NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

PERFORMANCE ANALYSIS OF MOBILE AD HOC NETWORKING ROUTING PROTOCOLS

by

Lee Kok Thong

December 2004

Thesis Advisor:	Geoffrey Xie
Second Reader:	Su Wen

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE *Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** December 2004	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE: Performance Analysis of Mobile Ad Hoc Networking Routing Protocols

5. FUNDING NUMBERS

6. AUTHOR(S) Lee Kok Thong

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

This thesis presents a simulation and performance evaluation analysis of the various routing protocols that have been proposed for the Mobile Ad Hoc Network (MANET) environment using the Network Simulator-2 (NS-2) tool. Many routing protocols have been proposed by the academic communities for possible practical implementation of a MANET in military, governmental and commercial environments. Four (4) such routing protocols were chosen for analysis and evaluation: Ad Hoc On-demand Distance Vector routing (AODV), Dynamic Source Routing (DSR), Destination-Sequenced Distance Vector routing (DSDV) and Optimized Link State Routing (OLSR). NS-2 is developed and maintained by the University of Southern California's Information Sciences Institute (ISI). Leveraging on NS-2's simulation capabilities, the key performance indicators of the routing protocols were analyzed such as data network throughput, routing overhead generation, data delivery delay as well as energy efficiency or optimization. The last metric is explored, especially due to its relevance to the mobile environment. Energy is a scare commodity in a mobile ad hoc environment. Any routing software that attempts to minimize energy usage will prolong the livelihood of the devices used in the battlefield. Three important mobility models are considered, namely, Random Waypoint, Manhattan Grid, and Reference Point Group Mobility. The application of these three models will enhance the realism of simulation to actual real life mobility in an urban or military setup scenario.

The performance of the routing protocols in varied node density, mobility speed as well as loading conditions have been studied. The results of the simulation will provide invaluable insights to the performance of the selected routing protocols. This can serve as a deciding factor for the U.S. Department of Defense (DoD) in their selection of the most suitable routing protocols tailored to their specific needs.

14. SUBJECT TERMS
Mobile Ad Hoc Networking, Routing Protocols, Network Simulation

15. NUMBER OF PAGES
153

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

PERFORMANCE ANALYSIS OF MOBILE AD HOC NETWORKING ROUTING PROTOCOLS

Lee Kok Thong
Civilian, Defence Science Technology Agency, Singapore
Diplôme d'Ingenieur, EPF, 1998
MSc (Telecommunication), King's College London, 1998

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 2004

Author: Lee Kok Thong

Approved by: Geoffrey Xie
Thesis Advisor

Su Wen
Second Reader

Peter Denning
Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis presents a simulation and performance evaluation analysis of the various routing protocols that have been proposed for the Mobile Ad Hoc Network (MANET) environment using the Network Simulator-2 (NS-2) tool. Many routing protocols have been proposed by the academic communities for possible practical implementation of a MANET in military, governmental and commercial environments. Four (4) such routing protocols were chosen for analysis and evaluation: Ad Hoc On-demand Distance Vector routing (AODV), Dynamic Source Routing (DSR), Destination-Sequenced Distance Vector routing (DSDV) and Optimized Link State Routing (OLSR). NS-2 is developed and maintained by the University of Southern California's Information Sciences Institute (ISI). Leveraging on NS-2's simulation capabilities, the key performance indicators of the routing protocols were analyzed such as data network throughput, routing overhead generation, data delivery delay as well as energy efficiency or optimization. The last metric is explored, especially due to its relevance to the mobile environment. Energy is a scare commodity in a mobile ad hoc environment. Any routing software that attempts to minimize energy usage will prolong the livelihood of the devices used in the battlefield. Three important mobility models are considered, namely, Random Waypoint, Manhattan Grid, and Reference Point Group Mobility. The application of these three models will enhance the realism of simulation to actual real life mobility in an urban or military setup scenario.

The performance of the routing protocols in varied node density, mobility speed as well as loading conditions have been studied. The results of the simulation will provide invaluable insights to the performance of the selected routing protocols. This can serve as a deciding factor for the U.S. Department of Defense (DoD) in their selection of the most suitable routing protocols tailored to their specific needs.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to dedicate this thesis work to my wife, Susie, who has been the most supportive person throughout my studies. She has constantly encouraged me even when things are not doing well. She took care of the family, and for her hard work and dedicated spirit, I would like to take the opportunity to give a big thanks to my beloved wife.

I would like to thank my thesis advisor, Prof. Geoffrey Xie, for his research advices, guidance, thesis support and encouragement. His thought-provoking questionings have made the thesis work both an interesting and challenging process, which I have enjoyed a lot. I would also like to extend my gratitude to my second reader, Prof. Su Wen, for her time and effort to assist me. I appreciate her mentorship.

I would like to thank the Defence Science Technology Agency (DSTA) of Singapore for sponsoring my studies at NPS as well as my superiors, Mr. Eugene Chang and Mr. Tan Ah Tuan, for supporting my scholarship. I would like to extend my appreciation to all the Professors in the Computer Science Department who have taught me and imparted their valuable knowledge and made this a memorable stay in this beautiful California city, Monterey.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. DEFINITION

A mobile ad hoc network (MANET) consists of mobile nodes such as computing devices like laptops and personal digital assistants (PDAs), that use wireless connections to link up to each other for the purpose of communication. These networks are generally dynamic collections of self-organizing mobile nodes with links that are characterized by dynamic topology changes and no fixed infrastructure. This is in contrast to the well-known single hop cellular network model that supports the needs of wireless communication by installing base stations as access points. In these cellular networks, communications between two mobile nodes completely rely on the wired backbone and the fixed base stations. In MANET, however, no such infrastructure exists and the network topology changes in an unpredictable manner since nodes are free to move.

The main communication medium is broadcast. Nodes can be regarded as wireless mobile hosts with a short-term power supply, a relatively short communication range, low processing power and limited bandwidth.

B . APPLICATION OF MOBILE AD HOC WIRELESS NETWORK

The recent rise in popularity of mobile wireless devices and technological developments have made possible the deployment of such networks for several applications. Indeed, because ad hoc networks do not have any fixed infrastructure such as base stations or routers, they can be quickly deployed regardless of the place and time since they are not hindered by the need for an infrastructure setup. As such, they have become highly applicable to emergency deployments, disasters, search and rescue missions and military operations.

1. Military Applications

When conducting tactical military operations in a foreign environment, seldom are there fixed supporting infrastructures for the different military units to exploit. Mobile ad hoc networks are extremely convenient for establishing not just voice, but also data and video communication. The essence of the deployment of a mobile ad hoc network is in its fast turn around time, which enables the military operations to be executed in the shortest time possible. It enables rear-end commanders to perform command and control functions by sending orders and tactics via these ad hoc networks to its front-end troops. The extensibility of the network, as well as its reliability, coupled with secure communications, will enable it to be a force multiplier for a modern military setup, especially valuable in the conduct of specific warfare such as surveillance and deployment of quick reactive forces.

2. Collaborative/Distributed Computing

In a commercial environment, ad hoc networks can provide individuals or groups of individuals to quickly and minimally establish a communication network that enables them to do collaborative and distributive work. It could be video conferencing between multiple parties located in different parts of the campus such as professors conducting lessons to students. The students do not have the geographical constraints and is free to roam around while receiving information. Business partners get together in a meeting and can quickly transfer files and data via an ad hoc network. There are endless applications to be exploited using a wireless ad hoc network to better network people.

3. Emergency Operations

In times of civilian emergencies, for example, the collapse of a building or localized chemical or biological contamination within an area, the formation of a mobile ad hoc network presents itself as an important tool that can help rescuers to police and manage the situation better. Different rescue entities can be equipped with portable devices, connected via ad hoc mode. They can communicate among each other in real time and provide updates to an ad hoc command post or call for backup between each other. In a natural disaster, the majority of the existing infrastructure would have been disabled or destroyed; the mobile ad hoc network can present itself as an excellent choice for a co-ordination tool between the different emergency response teams.

C. OBJECTIVE AND SCOPE

The objective of this thesis is to study and analyze the performance of four routing protocols for mobile ad hoc network environments using an open-source network simulation tool call *network simulator* [NS2]. These four routing protocols which are being investigated, are Ad Hoc On-demand Distance Vector routing (AODV) [Perkins 1999], Dynamic Source Routing (DSR) [Johnson 2000], Destination-Sequenced Distance-Vector routing (DSDV) [Perkins 1994] and Optimized Link State Routing

(OLSR) [OLSR 2003]. The simulation environment will be conducted with the Linux operating system whereby it is possible to experiment with the impact on these routing protocols in different node mobility conditions, loading, mobility models as well as node density. The performance analysis will focus on network overhead, data throughput, communication delay in addition to energy consumption of the four routing protocols. Moreover, the applicability and suitability of the routing protocols in urban and military deployment setup scenario will also be considered. The simulation will take into consideration the constraints that are experienced by military operations and the environment.

In essence, the thesis endeavors to answer the following questions:

(i) How does table-driven protocols (OLSR, DSDV) perform compared to on-demand routing protocols (AODV, DSR) under different mobility models? Currently, there are many ongoing research investigations using Random Waypoint [Maltz 1996] [Camp 2002] as the mobility model. Few have considered using other mobility models such as the Manhattan Grid mobility model [Tech 1998] and Reference Point Group mobility [Hong 1999] [Camp 2002]. Chapter IV will examine each of these models in detail . Simulations will be conducted using these models in this thesis. In addition, situations such as nodes' speed, network loading as well as the node density are considered. The metrics used to compare these four routing protocols include packet delivery ratio, network delay, routing overheads and energy consumption.

(ii) OLSR is a relatively new protocol compared to AODV, DSR and DSDV. This thesis attempts to explore the possibility of improving OLSR performance by tweaking, for example, its *hello intervals* and *topology control intervals* parameters. Currently, OLSR is still in an experimental stage and the IETF's Request For Comments (RFC) number for OLSR is 3626 [RFC 3626]. Default values for OLSR parameters are proposed in the RFC, these parameters will be investigated to ascertain whether these parameters provide an optimal network performance to OLSR.

D. THESIS OUTLINE

Chapter II begins with an introduction to the current existing routing protocols that are either ready for deployment in mobile routers or are in an academic experimental stage. The routing protocols will be broadly classified as on-demand and table-driven. Other classifications exist and will be briefly discussed. The third chapter will specifically focus on the optimized link state routing given its acceptance for deployment by some military product vendors. Chapter IV will be dedicated to the simulation setup and usage limitation of this simulation software. Chapter V will discuss the results of the simulation. Finally, Chapter VI will conclude these studies and recommend further actions and propose future study areas.

II. MOBILE AD HOC ROUTING PROTOCOLS

A. ISSUES IN DESIGNING ROUTING PROTOCOLS

The traditional routing protocols that have been used in the design of a wired network cannot be applied directly to a wireless mobile ad hoc network due to the highly dynamic nature of mobile nodes as well as the non-existence of a central authority for overall control.

The major challenges facing the design of mobile ad hoc routing protocols are the node's mobility, resource constraints such as power and bandwidth as well as unstable channel states.

Due to the nature of mobile nodes, which can be highly dynamic, communication between mobile nodes is often characterized by frequent path breaks and reconnections. These disruptions are less common in wired environments whereby routers are typically housed in racks and locked up in computer rooms. As such, it is possible to imagine that traditional routing protocols, such as Routing Information Protocol (RIP) [RIP RFC] or Open Shortest Path First (OSPF) [OSPF RFC], are not suitable candidates for MANET routing protocols.

In addition to mobility, the power availability to the mobile node is also a serious consideration. Unlike typical wired-link routers, the power source of mobile nodes come from non-permanent power sources such as batteries. As such, the power usage by routing protocols will have an impact on the overall performance of the network. Imagine the case where a node is the sole router linking two independent networks, any unnecessary usage of power on this node will further drain power from it and thereby cause a link breakage between the two networks when the node runs out of power. Besides power, bandwidth is also a scare commodity in a MANET environment. Traditional routers and switches have reached the state of fast ethernet bandwidth (100Mbps) or even gigabit Ethernet rates (1000Mbps). The wireless connectivity rate is no where near these rates. While current 802.11a technology allows for a theoretical transfer rate of 11Mbps, faster wireless access rates of 54Mbps (802.11g) can be achieved today for static wireless devices connecting to the base station infrastructure.

However, the practical transfer rate of wireless connectivity is more often in the region of 10-40% of the theoretical capability [Througput] at close range. Operating under hasher conditions will rapidly decrease the throughput, especially when there are many obstacles blocking the communication path between the nodes.

The broadcast nature of radio channels can be highly unstable, especially when a mobile node is on the move and it also presents a time-variant characteristic. As such, layer 3 routing protocols have to interact with layer 2 MAC protocols to search for available channels when none are found. When there is simultaneous transmission (at the MAC layer), packets do collide. A receiver can receive simultaneous data from different senders, which are totally out of range from each other. As such, they do not know that different parties are sending data to the sender at the same time. As the number of nodes increase, this problem can be aggravated.

Other issues include limited physical security for mobile ad hoc nodes. Generally, since the nodes are not statically located, they are prone to more physical security threats than fixed routers. Compromised nodes may pose serious problems to the entire network, it is possible to use them especially as devices to deviate data traffic or launching pad for attacks against other nodes.

For the military networks, typical military operations can cover large distances that result in the large scale deployment of mobile nodes or high-speed nodes such as mobile routers mounted on tanks or unmanned vehicles. As such, a good routing protocol in this case should be scalable and robust enough for rapid building and tearing down of routes.

B. ROUTING PERFORMANCE ISSUES

The following lists some criteria that can be used in the design consideration of routing protocols using quantitative and qualitative metrics.

The quantitative metrics include the following.

1. Throughput

Throughput can be defined as the overall percentage of data received over the data sent in a closed system for a specific period of time. Statistical measures can be used to analyze throughput. This is a fundamental measure of the performance of a network, and therefore, an important factor to consider.

2. Delay

The delay is the overall time taken from the moment the data is transmitted to the moment it is received by the designated destination. Delay affects applications in many ways. Applications that are delay-sensitive such as video streaming and voice cannot function properly when there is a long delay.

3. Efficiency

Protocol efficiency is the measurement of the routing effectiveness. To achieve the same throughput between two protocols, it might be necessary to expend more routing overheads than another or there are built-in buffer requirements to allow for the temporary storage of data. Also, the ratio of control bits over the overall data sent can be used as a gauge of the protocol efficiency.

The qualitative measurements of routing performance can include the following.

4. Loop Freedom

Network protocols can resolve the issue of infinite looping by using time-to-live (TTL) features that are traditionally done in IP networks. It would be greatly beneficial for the network as a whole if loop freedom can be *avoided* rather than resolved. Loop-free routing protocols generally will allow for better performance ad hoc networks.

5. Traffic-Aware Routing

The traffic distribution in ad hoc networks (and even in traditional networks) is uneven. It is time-dependent and application-dependent. As such, routing algorithms that can intelligently do load balancing by using resources evenly can prolong the life span of these mobile nodes.

6. Power Mode

Not all nodes need to be active all the time since not all nodes participate in routing at all times. Nodes that do not take part should be able to go into the "sleeping" mode to reduce power usage. This is especially true when multiple routes exist between nodes and some nodes can be temporarily “turned off” without too much impact on the overall performance.

C. TABLE-DRIVEN ROUTING PROTOCOLS

Table-driven routing protocols for Mobile ad hoc networks are proactive in nature. They constantly maintain routing tables of the entire topology of the network. They exchange routing information to obtain the latest “snapshot” of the topological information. This results in less look-up time for the route path to reach a specific destination. However, to achieve such a shorter delay, table-driven protocols have to pay the price of sending periodic control messages even though the nodes may not be transmitting to each other. In some cases, this large amount of data control message may be detrimental to a low-bandwidth MANET network.

1. Destination-Sequenced Distance Vector (DSDV)

The first table-driven protocol to be considered is Destination-Sequenced Distance-Vector Routing (DSDV) [Perkins 1994]. It is a routing algorithm based on the Bellman-Ford algorithm, a distance vector type of routing protocol. It improves on the Bellman-Ford algorithm by making sure it is free of loops. This is accomplished by assigning each route a unique sequence number. This distinguishes new routes from old routes, preventing the formation of loops based on old routing data. More specifically, each node has a table, which consists of a destination, a route, a hop count and a sequence number, which is how the routing information is stored and accessed in the DSDV protocol. Updates can be distributed via two methods. The first is done through a *full* dump, which means that the entire routing table that a given node has is sent to its directly attached neighbors. While providing quick convergence when the network is first being set up, this is a large amount of data to be continually sent if the network is not changing much. A second kind of update is “*incremental*” which, as its name implies, only sends information about the difference in routes between the current table and the last full dump sent. In order to keep tabs on the route broadcasts sent, each new route broadcast contains a sequence number unique to the broadcast, in addition to the route sequence number. When updates are received, the route with the most recent sequence number is always added to the routing table (or kept in the routing table). If two routes have the same sequence number, hop count is used as the deciding factor instead.

Another feature of DSDV is that it delays the broadcast of routing information based on the average settling time for the network. This avoids the sending of extra updates if an improved route will be arriving in the near future.

Some issues exist for DSDV; one of which is route fluctuation. Due to its criteria of route updates, where routes are preferred if the new sequence number is higher or the same as the existing sequence number, and in some cases, routes between two specific nodes can change back and forth. This partially results from nodes that transmit their routing updates independent of each other. Another problem is related to the assumption that DSDV assumes that all network links in a MANET environment are bi-directional. This may not be the case, sometimes, due to environmental limitations. Only uni-directional links exist between two nodes. From Figure 1, it is possible to see that even with uni-directional links, data can be routed from point A to C. However, DSDV would falsely consider the destination as unreachable.

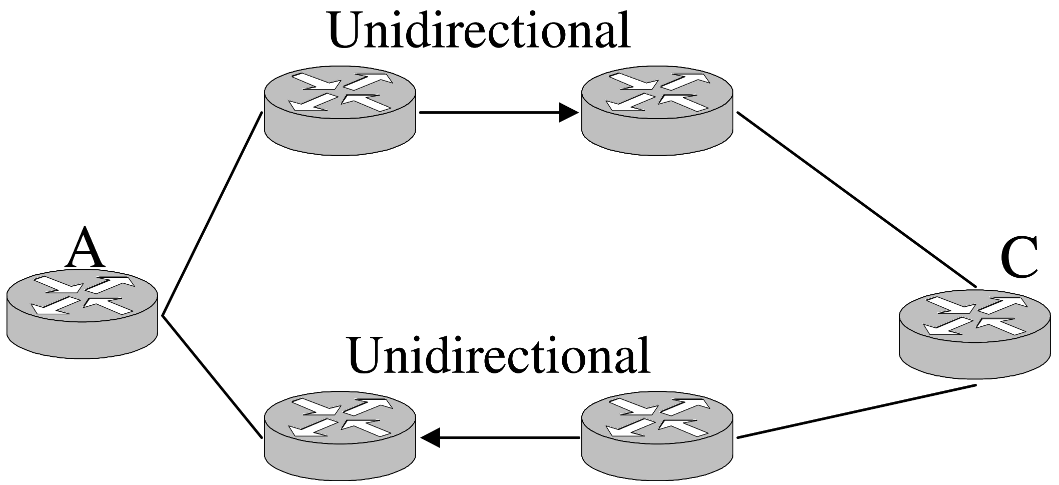


Figure 1. Ad hoc network with uni-directional and bi-directional links

Moreover, DSDV has excessive communication or routing overhead due to periodic and triggered updates. Its commercial implementation is rare. At the time of this thesis research, one known DSDV simulator that has been developed is the NS-2 [NS2].

2. Optimized Link State Routing (OLSR)

Given the adoption of OLSR as the routing protocols by some vendors such as Inter-4 [Inter] and Rugged Notebooks [Rugged], both of which are selling tactical PDAs equipped to work in a mobile ad hoc network to mainly defense contractors, a comprehensive analysis has been dedicated to OLSR specifically in the next chapter.

3. Cluster-Head Gateway Switch Routing (CGSR)

A similar proposed routing protocol to DSDV is the Cluster-head Gateway Switch Routing (CGSR), which uses a hierarchical routing address space instead of a flat address space [Chiang 1997]. The protocol describes a process for electing "cluster heads" within the network that act as the focal point of activity within that part of the network. When two cluster heads come within contact or when a node moves out of the range of any existing cluster head, this causes a change in the cluster head assignment. Other than using a different addressing scheme, CGSR is similar to DSDV. Each node keeps a "*cluster member table*", which lists each mobile node in the network and its associated destination cluster head. The DSDV algorithm can be used for route propagation. A separate routing table is kept in addition to the cluster member table. To forward a packet, a node first looks up the destination in the cluster member table and routing tables to find the nearest cluster head along the route to the destination, then checks the routing table to figure out the next hop to reach the intended cluster head. While the hierarchical addressing does make this routing protocol more scalable, additional latency is created by having to elect cluster heads periodically when the network changes. In addition to this problem, because the route selection is between the cluster heads, the path taken to reach its destination may not be necessarily optimal.

In summary, Table 1 summarizes the different characteristics of the table-driven protocols.

Parameters	DSDV	CGSR	WRP
Time Complexity (link addition / failure)	O(d)	O(d)	O(h)
Communication Complexity (link addition / failure)	O(x=N)	O(x=N)	O(x=N)
Routing Philosophy	Flat	Hierarchical	Flat
Loop Free	Yes	Yes	Yes, but not instantaneous
Multicast Capability	No	No	No
Number of Required Tables	Two	Two	Four
Frequency of Update Transmissions	Periodically & as needed	Periodically	Periodically & as needed
Updates Transmitted to	Neighbors	Neighbors & cluster head	Neighbors
Utilizes Sequence Numbers	Yes	Yes	Yes
Utilizes "Hello" Messages	Yes	No	Yes
Critical Nodes	No	Yes (cluster head	No
Routing Metric	Shortest Path	Shortest Path	Shortest Path

Table 1. Comparison of Table-driven protocols' characteristics (From: [Royer 1999])

D. ON-DEMAND ROUTING PROTOCOLS

A completely different approach from table-driven routing is source-initiated on-demand routing. The source node initiates the routing request whenever there is a need to transmit data to a destination. The routing process commences with a route discovery process within the network. Once a route is found or all possible route permutations have been examined, the routing process is completed. A route maintenance procedure is necessary to keep the active route alive until either the destination becomes inaccessible along every path from the source or until the route is no longer desired.

1. Ad Hoc On-Demand Distance Vector (AODV)

In AODV [Perkins 1999], the only nodes that participate in the entire routing process are those sitting in the direct path between the source and destination node. Hence those nodes that do not lie on active paths neither maintain any routing information nor participate in any periodic routing table exchanges. Thus AODV seeks to minimize the number of control messages sent between the nodes.

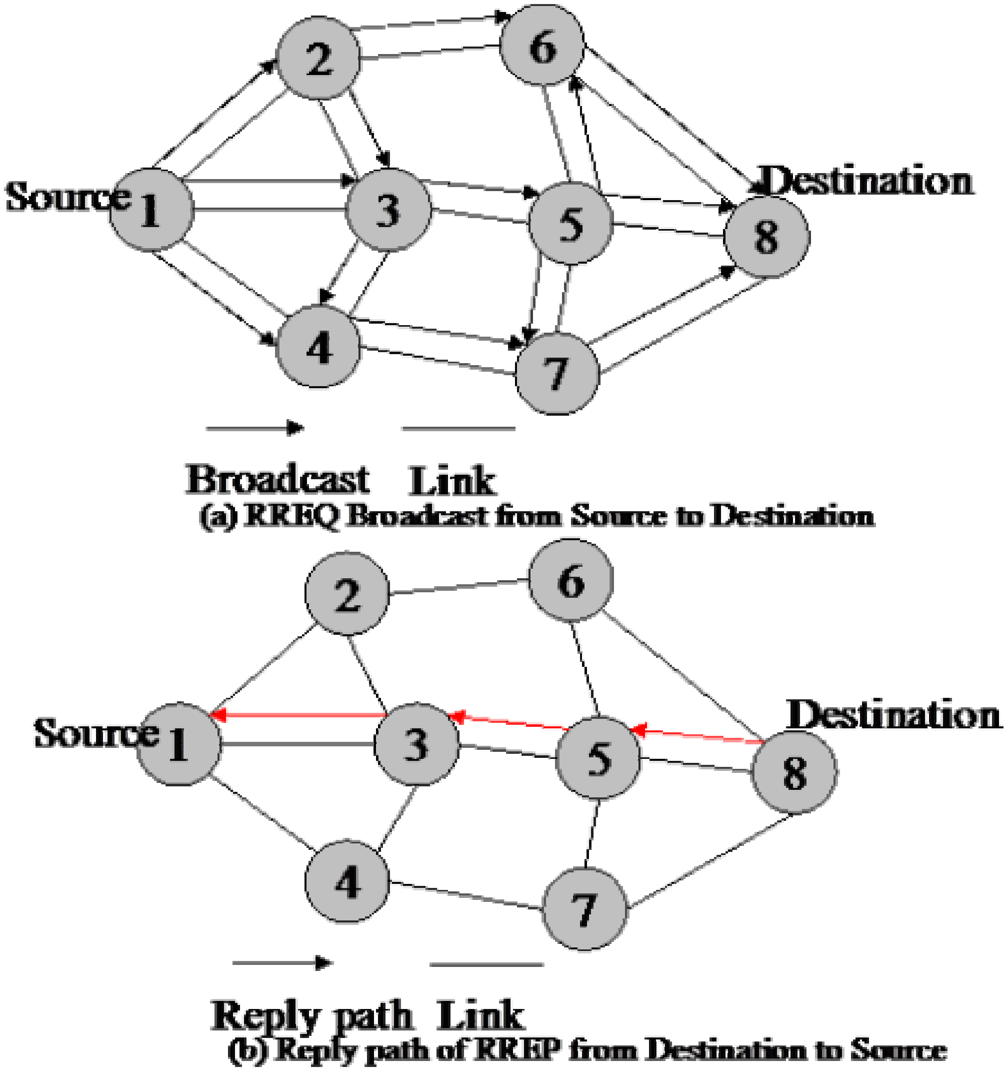
Mobile nodes can make use of "hello" messages to become aware of the other nodes in the neighborhood. "Hello" messages are broadcast type traffic. The routing tables of the nodes within the neighborhood are organized to optimize response time to local movements and provide quick response time for requests for the establishment of new routes. The algorithm's primary objectives as stated in [Perkins 1999] are:

- To broadcast discovery packets only when necessary
- To distinguish between local connectivity management (neighborhood detection) and general topology maintenance
- To disseminate information about changes in local connectivity to those neighboring mobile nodes that are likely to need the information.

AODV borrows the concept of DSDV with the aim of reducing the need for system wide broadcasts as much as possible and AODV improves it by using a "monotonically increasing number" for the destination sequence number to replace old and stale routes, the result of which is a loop-free, highly situational responsive and bandwidth-efficient routing protocol. AODV is capable of both unicast and multicast routing.

In short, if A needs a route to B, it broadcasts a ROUTE REQUEST message. In each ROUTE REQUEST (RREQ), a pair of information, namely the source address and the broadcast identification number, is unique. Each node that receives this request message, and does not have a route to B, rebroadcasts it. The nodes along the routing path also keep track of the number of hops the message has made, as well as remembering who sent it the broadcast. If a node has the route to B, it replies by unicasting a ROUTE REPLY (RREP) back to the node from which it received the request. The reply is then forwarded back to A by unicasting (based on prior broadcast information) it to the next hop towards A. This establishes a uni-directional route (asymmetrical link). For a bi-directional route(symmetrical link), this procedure will need to be repeated in the reverse direction. To achieve faster convergence in the network, and thus higher mobility, a ROUTE ERROR message can be broadcast to the network in the case of a link breakage. Hosts receiving the error message remove the route and re-broadcast the error messages to all nodes, with information added about new unreachable destinations.

Figure 2 illustrates the discovery process of AODV.



(a) RREQ Broadcast from Source to Destination

(b) Reply path of RREP from Destination to Source

Figure 2. Route Discovery process of AODV

AODV scales better than DSDV given that fewer control overheads are generated. As such, for the large-scale deployment of ad hoc networks, AODV will perform far better as far as scalability is concerned. However, the tradeoff in so minimizing route updates is that there is considerable delay in the acquisition process of the best route to reach the destination. Table-driven protocols have no such problem since the routes already exist in every node. This is especially aggravated in the case where the diameter of the network is large and applications used are delay-sensitive such as video streaming.

2. Dynamic Source Routing (DSR)

Dynamic Source Routing (DSR) is a reactive routing algorithm based on link-state routing and it was first proposed by [Johnson 2000]. It is based on the concept of source routing. Routes caches are kept at the mobile nodes so as to enhance the discovery process. These caches are also continuously updated throughout the process. DSR allows for packets to travel over a different route from source to destination than from destination to source. Given this flexibility in DSR, each sender can choose its optimal path to reach its destination, thereby achieving some sort of load balancing and making the data transfer process more robust.

Two major phases take place in DSR: route discovery and route maintenance.

In *route discovery*, the sender floods the network with RREQ messages (including source IP address, destination IP address and an unique request ID) and nodes receiving the flood message will forward the RREQs after appending their names onto the RREQs. The destination node receiving the final RREQ will unicast a RREP back to the sender node. Each node will include its identification into the list of addresses that constitute the path from source to destination. See Figure 3.

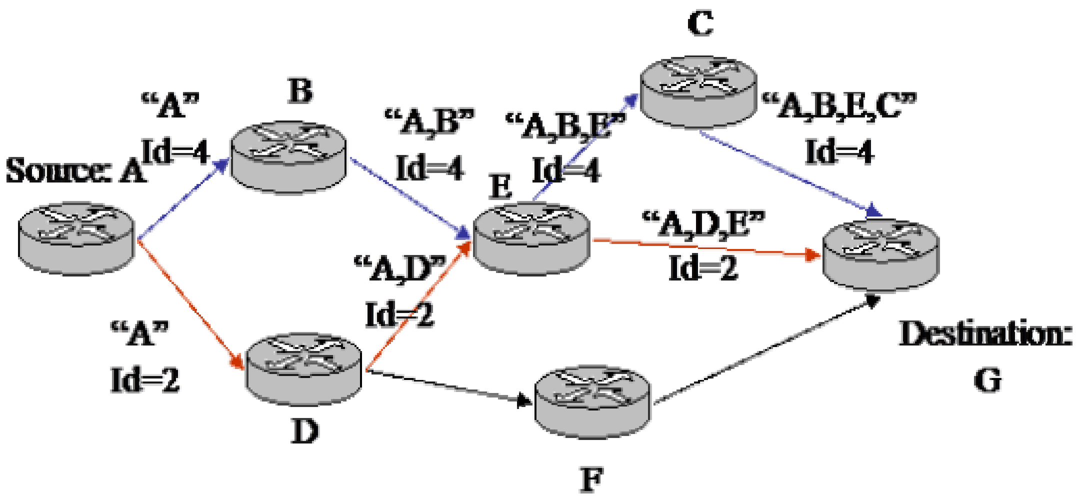


Figure 3. DSR Route Discovery

In *route maintenance*, the maintenance is achieved through means of two types of control packets, i.e., route error and acknowledgements. Once there is a data-link failure, a route error message is generated. Upon receipt of the route error packet, the hop in error is removed from the route cache and all routes using this hop will be truncated. A rediscovery process is necessary to establish alternate paths. Acknowledgement packets are used to ensure the correct functioning of the links between the nodes. An example would be nodes could eavesdrop onto other nodes' transmission when they transmit data, which can help indicate if the transmission process is successful.

Once the maximum number of re-transmission is reached, and no receipt confirmation is received, a node will return a ROUTE ERROR message to the original sender of the packet, identifying the link over which the packet could not be forwarded. Whenever any intermediate node, receiving the RREQ, knows of the full path (using its route cache) to the destination, it will send a RREP message (on behalf of the destination) to the originator and the RREQ broadcast would stop here.

DSR potentially has a larger overhead and is intended for moderate speed (with respect to the packet transmission latency and transmission range) mobile nodes and is not scalable to very large networks. For smaller network sizes, DSR will be able to adapt quickly to dynamic topological changes. Moreover, loop freedom is guaranteed. It supports asymmetric links and allows nodes to keep multiple routes to one destination in their route cache, and hence, will be able to achieve faster route recovery. However, like AODV, there will be delay due to set-up time for the route path.

DSR allows for support of seamless interoperation between an ad hoc network and the Internet, allowing packets to transparently be routed from the ad hoc network to nodes in the Internet and from the Internet to nodes in the ad hoc network [Broch 1999b]. To achieve this, a *gateway* node, capable of understanding the dual networks, is created to participate in routing between both networks.

One of the tricky problems that DSR addresses is that wireless links are not always symmetrical because of discrepancies in transmission power, receiver sensitivity and propagation patterns. In addition, the entire selected path is actually propagated together with the request message. The same is true for route maintenance, error messages. In addition, DSR does not require any periodic updates or keep-alive messages from nodes. This helps to reduce overhead in routing and conserves scarce bandwidth.

Experiments [Johnson 2000] have shown that higher nodal density has led to a better overhead efficiency (ratio of overheads to actual useful data payload) . However, as the mobile nodes become more dynamic in motion, the overhead will increase. The discovered routes have been shown to be near to optimal route length.

3. Temporally-Ordered Routing Algorithm (TORA)

In TORA, routes are defined by a Directional Acyclic Graph (DAG) [Gaf 1981], [Cor 1995] rooted at the destination node. To create DAG, nodes will use a height metric, consisting of five parameters: logical time of link failure, unique ID of node defining the new reference level, reflection indicator bit, a propagation ordering parameter with respect to common reference level, and unique ID of node. These five parameters are highlighted in the Figure 4 and 5, indicated by the brackets. Three types of control packets are used: query (QRT), update (UPD), clear (CLR). QRT messages are flooded to all intermediate nodes until the destination node is reached and upon which a UPD message is used to update nodes along the reversal path from destination to source. UPD messages are also used to indicate link failure. A CLR broadcast is sent throughout the network to clear invalid routes. Figure 4 shows the connecting nodes and their heights after QRT and UPD messages have flooded the network and a path is found.

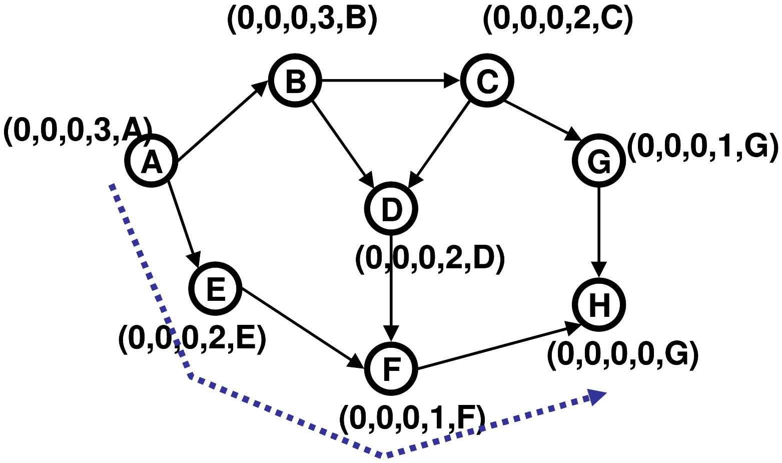


Figure 4. Establishment of DAG for TORA

In 3-dimension, it is possible to imagine the “height” of source being taller than that of the destination and the flow of data/route will be analogous to that of water flowing down from a higher to lower ground. The process of establishing the DAG is similar to the query and reply process as proposed in a light-weight mobile routing (LMR) [Corson 1995]. Upon link failures, shown in Figure 5, route maintenance is necessary to re-establish the DAG rooted at the same destination. As shown in Figure 5(b), the link failure at D generates a new reference level, resulting in a propagation of reaction of link reversal, which effectively reflects the changes in adaptation to the new reference. The effective new DAG is shown in Figure 5(d) with the isolated, disconnected B-C-D network.

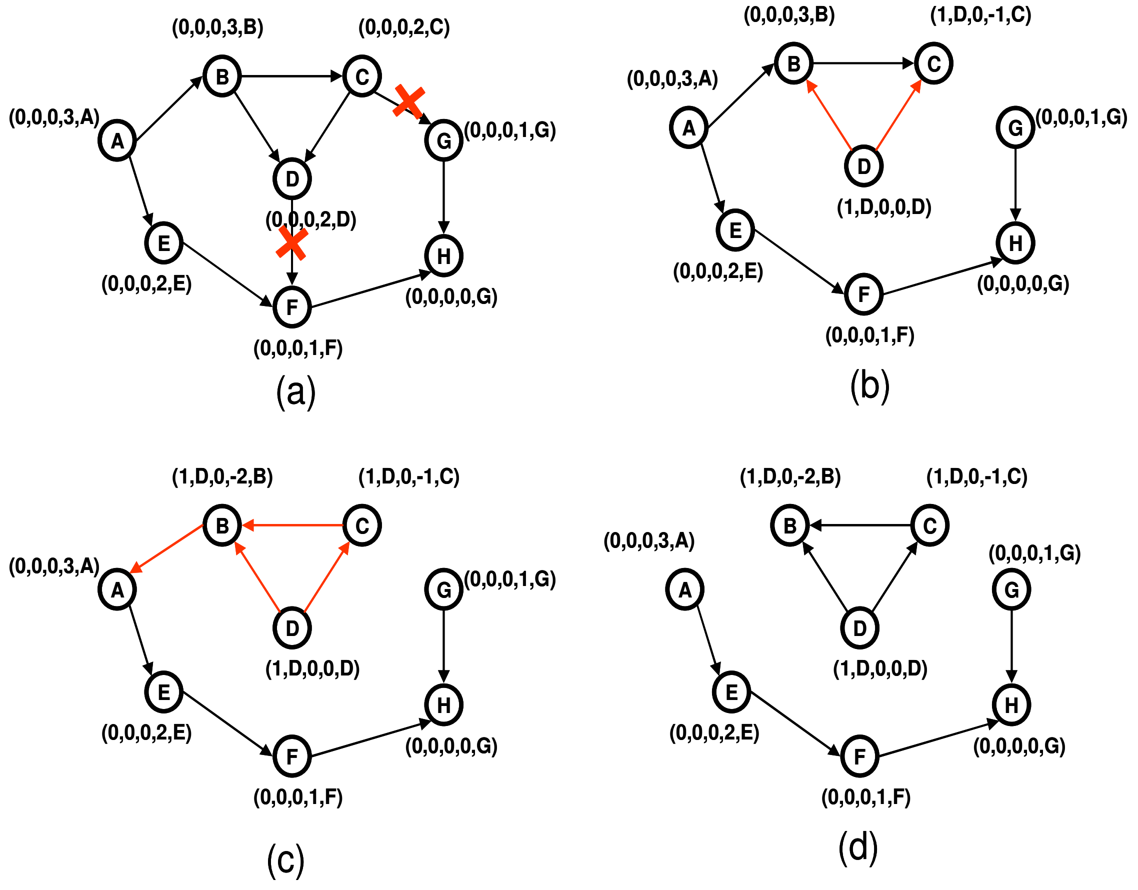


Figure 5. Route Maintenance for TORA/link reversal process

As timing is an important factor within the height metric, synchronization of timing is important for effectively executing TORA routing. This is sometimes achieved through an external clock source such as GPS. However, not all mobile devices are GPS-enabled, and therefore, this routing protocol will pose a considerable challenge for wide-spread deployment and inter-operability for heterogeneous mobile devices.

4. Associative-Based Routing (ABR)

Proposed in [Toh 1996], Associativity-Based Routing(ABR) provides yet another approach towards a bandwidth-efficient routing protocol. ABR is a source-initiated, reactive routing algorithm.

The author also believes that many nodes spend most of their time doing their own work locally and relatively less time in communicating with other nodes. Hence, there is no need to set up routes to inactive nodes, at least for the period when they do not participate in any communication with the source node.

The term "*degree of association stability*" [Toh 1996] has been used as a metric in ABR. In ABR, mobile nodes are said to be *highly mobile* when they have low *associative ticks* with their neighbors. Conversely, a highly stable mobile node would have high associative ticks associated with it and it would be a preferred node for the routing of information. The routing metrics employed for determining the associativity ticks are 1) longevity of a route and 2) relaying load of intermediate nodes supporting existing routes. All nodes generate periodic beacons to indicate alive status. When a neighbor node receives a beacon, it increases its associativity tick with respect to the sending node in the associativity table. Associativity ticks are reset when the neighbors of a node or the node itself moves out of proximity. See Figure 6 for ABR route establishment.

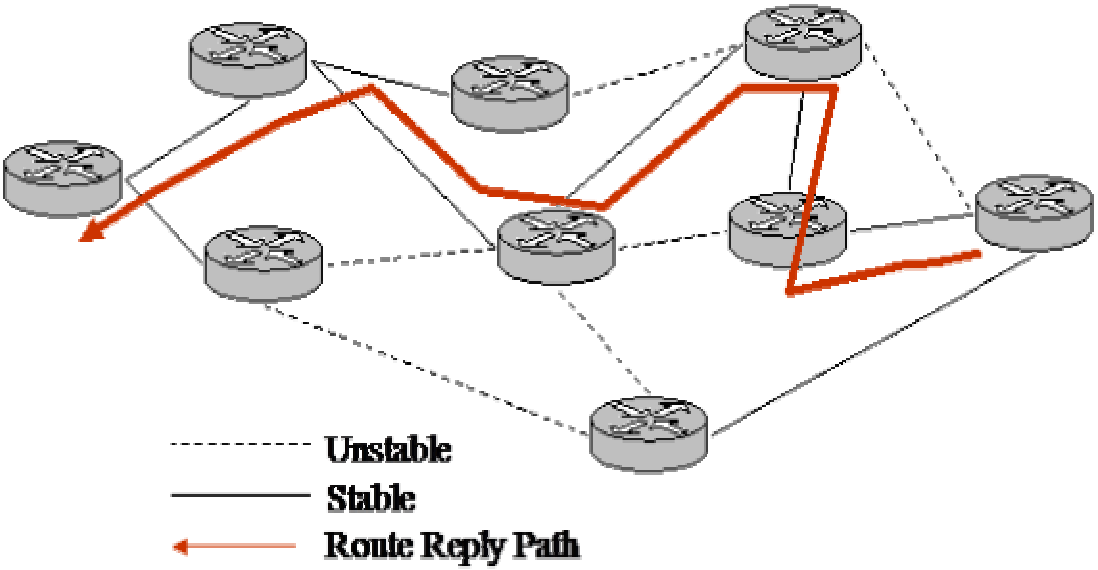


Figure 6. Route Establishment for ABR

In a route discovery process, the source broadcasts a QRY message for searching the destination node and each intermediate node appends their address and associativity ticks to the QRY message. If the message is received before, the node simply discards it. The destination can then examine the associativity ticks of potentially multiple possible paths to select the optimal route. If the multiple paths have the same overall degree of stability, it will use the route with the minimum number of hops as the tie-breaker.

At times when there is a change in the network topology, a route-reconstruction-(RRC) phase is initiated to reconstruct a new routing table. This phase consists of partial route discovery, invalid route erasure, valid route update and new route discovery. RRC can be initiated by several nodes such as the source, destination or intermediate nodes. In the case of the destination node, it will notify its neighbors of its movement and the possibility of changed routes. A sequence number is used to keep track of the "freshness" of the RRC so that an older RRC will be deleted.

When the route is no longer desired, the source may not be aware of any route node changes because of any partial reconstruction within the route. The source node initiates a route delete (RD) broadcast to erase the invalid route and the broadcast message received by the intermediate nodes will be updated in their routing tables.

ABR seeks to achieve long-lived routes through the better use of time and space in a MANET environment, the result of which is lesser route reconstruction, and hence, higher attainable throughput for data transmission. However, the path chosen may not necessarily be optimal in the selection process. The stability of the node linkages has higher priority. Moreover, another disadvantage may be that a route cannot be established due to unavailability of stable signal paths, thus denying the establishment of connectivity altogether.

In summary, Table 2 compares the features of the on-demand routing protocols [Royer 1999].

Performance Parameters	**AODV**	**DSR**	**TORA**	**ABR**
Routing Philosophy	Flat	Flat	Flat	Flat
Loop Free	Yes	Yes	Yes	Yes
Multi-cast support	Yes	No	No	No
Routes maintain in	Route Table	Route Cache	Route Table	Route Table
Routing Metric	Freshest and Shortest Path	Shortest Path	Shortest Path	Associativity and Shortest Path
Route Reconfiguration Methodology	Erase Route; Notify source	Erase Route; Notify source	Link Reversal; Route repair	Localized Broadcast Query

Table 2. Comparison of Characteristics of On-demand Routing Protocols (From: [Royer 1999])

E. COMPARISON OF TABLE-DRIVEN AND ON-DEMAND

Table 3 summarizes the main differences between table-driven and on-demand classes of routing protocols.

	Table-driven	**On-demand**
Availability of Routing Information	Immediately from route table	After a route discovery
Route Updates	Periodic broadcast Advertisements	As per request
Routing Overhead	Increases as the size of the network increases and it is independent of network traffic	Increases as the number of mobile nodes are added and also increases with faster node mobility

Table 3. Main Characteristics Difference between On-demand and Table-driven Protocols

F. HYBRID ROUTING PROTOCOLS

Routing in a versatile environment, such as MANET encounters, is an extremely challenging task. Certain protocols excel for specific types of ad hoc networks, still, for a single basic protocol, it may not be able to perform as well over the entire space of ad hoc networks. For this reason, hybrid routing protocols have been designed to conform to any arbitrary ad hoc network. However, their performance evaluation and overall implementation in practical situations is still an on-going process.

1. Zone Routing Protocol (ZRP)

As highlighted in the above paragraphs, conventional table-driven and on-demand ad hoc routing protocols each have their pros and cons. Zone routing protocol (ZRP) [Haas 2002] attempts to use the advantages in each of the class of routing protocols and thereby uses the proactive nature of table-driven protocols to discover neighbor nodes in the vicinity of a group (Intra-zone routing) and the passive nature to disseminate routes to different groups on a per-request basis so as to minimize the route exchanges between groups (Inter-zone routing). As such, some may consider ZRP as a framework or strategy for which it is possible to use other routing protocols rather than being a routing protocol itself. The IETF RFCs for ZRP did not specify which inter-zone or intra-zone routing protocols to be used for deployment. Choosing from existing on-demand and table-driven routing candidates is an on-going research area.

ZRP helps to reduce traffic generated compared to pure proactive or reactive routing. Since proactive updates are propagated only locally within a zone, the amount of control traffic does not depend on network size. The reactive routing is more efficient than flooding since local topology information can be used. Moreover, ZRP is able to identify multiple routes to a destination, which provides better reliability and performance. ZRP routes are free from loops. Unlike hierarchical protocols [Pearlman 1999], ZRP is a flat protocol that can reduce congestion and overhead. Generally, ZRP is targeted for large scale networks [Das 2000].

2. Landmark Routing with Group Mobility (LANMAR)

Proposed by Pei and his group from the University of California, Los Angeles, Landmark ad hoc routing with group mobility (LANMAR) [Pei 2000] combines the features of Fisheye routing protocol [Gerla 2000] and Landmark routing [Tsuchiya 1988] to achieve a more efficient routing protocol. The idea is to extend Fisheye routing by grouping all the routes to reach the group members and sending only a single route information to reach the landmark. This protocol has proven, by means of simulation, to be scalable for large scale networks. LANMAR has been shown to be more efficient in terms of throughput and overheads when compared to AODV and DSR when the traffic is medium to high load.

3. Sharp Hybrid Adaptive Routing Protocol (SHARP)

Optimal routing protocol can rely on network characteristics and adapt dynamically to the environment in which the MANET is operating. Sharp Hybrid Adaptive Routing Protocol (SHARP) [Ramasubramanian 2003], developed by the Cornell University, seeks to find an optimization point between proactive and reactive routing by dynamically adjusting how route information should be disseminated according to the network situation. The routing layer using SHARP protocol will optimize using a different metric, such as overhead, latency or jitter, for routes targeting that node. In general, SHARP can provide an informed, analytically-driven mechanism to find the optimal mix of proactive and reactive routing within a network.

G. OTHERS

In recent years, more researchers are looking into the security aspects of the routing, routing based on certain security features and the impact of security on routing performance. A number of notable security-based or secured routing protocols for references are discussed as follows.

1. Security-Aware Routing Protocol (SAR)

Security-Aware ad hoc Routing (SAR) [Yi 2001] proposes the incorporation of security attributes as metric parameters into ad hoc route discovery.

2. Secured Ad Hoc On-Demand Routing Protocol (S-AODV)

This is an extension of the existing AODV that takes into consideration Layer 3 security [Guerrero 2001].

3. Secure Routing Protocol (SRP)

Secure Routing Protocol (SRP) [Papa 2002] is adapted for DSR using *symmetric* crypto techniques (although the author did not preclude the use of PKI, if such a structure exists).

4. Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks (SEAD)

Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks [Hu 2003] is an extension to the DSDV.

5. Secure On-Demand Routing Protocol - ARIADNE

Ariadne [Perrig 2002] has proposed to prevent attacks originating from compromised nodes from tampering with uncompromised nodes and it also prevents other denial-of-service attacks in MANET.

THIS PAGE INTENTIONALLY LEFT BLANK

III. OPTIMIZED LINK STATE ROUTING PROTOCOL

The Protean Research Group [NRL] of US Naval Research Laboratory has developed an inhouse version of Optimized Link State Routing (OLSR) protocol, called nrlolsrd that can run on both UNIX and Windows platforms. Given the general acceptance of OLSR by the research group, a chapter has been dedicated for more detailed understanding of this routing protocol.

A. GENERAL INTRODUCTION

The Optimized Link State Routing protocol [OLSR 2003] is a proactive link state routing protocol. OLSR is explained in IETF's RFC 3626 [RFC 3626] and it is largely still in the experimental phase. There are two types of control packets used in OLSR: *Hello* packets and *Topology Control* packets (TC).

1. Neighborhood Discovery

Hello packets are used to build the neighborhood of a node and to discover the nodes that are within the vicinity of the node and hello packets are also used to compute the multipoint relays of a node. OLSR uses the periodic broadcast of hello packets to sense the neighborhood of a node and to verify the symmetry of radio links. The Hello messages are received by all one-hop neighbors, but are not forwarded. For every fixed interval, known as Hello Interval, the nodes will broadcast hello messages. Hello messages also allow the nodes to discover its two-hop neighbors since the node can passively listen to the transmission of its one-hop neighbor. The status of these links with the other nodes in its neighborhood can be either asymmetric, symmetric or multipoint relay (MPR) (see below for a more detailed explanation of MPR flooding). A symmetric link means that connectivity is bi-directional whereas asymmetric links are uni-directional. Given the set of one-hop and two-hops neighbors, a node can then proceed to select its multipoint relays, which will enable the node to reach out to all the neighbors within a two-hop range. Every node *k* will keep a *MPR selector set*, which contains all the nodes that has selected node *k* as a MPR. Hence, node k can only re-broadcast messages received from the nodes found in the MPR selector set [OLSR 2003].

2. Topology Dissemination and Routing Table Calculation

Topology control (TC) messages contains the MPR selector set information of a particular node *k*. These TC messages are broadcast periodically within the TC interval, to other MPRs, which can further relay the information to further MPRs. Thus, any nodes within a network can be accessed either directly or through the MPRs. With the neighborhood and topological information, nodes can construct the entire network routing table. Routing to other nodes is calculated using the shortest path algorithm such as Dijkstra's algorithm. Sequence numbers are used to ensure that the routing update information is not stale. Whenever there are changes to the topology or within the neighborhood, the MPR set is re-calculated, updates are sent to the entire network so that the routing has to be re-calculated to update the route information to each known destination in the network.

B. FULL FLOODING VS MULTIPOINT RELAYS

As specified above, hello messages are exchanged only between nodes one-hop away. Since the size of the MANET can be considerable, there is a need for a more efficient way of disseminating topological information. The traditional method would be pure full flooding into the network. While simple in implementation, it is not efficient since a great many control overheads are generated and not all are useful. Since a node within a network can be reached via many nodes (within the radio transmission range), only one node is necessary to transmit the message to it instead of multiple copies of the same message. MPR is a concept designed to reduce these control overheads by allowing *selective* flooding to occur. Only selected MPR nodes are allowed to re-broadcast topological information.

See Figure 7 for the comparison of pure flooding and selective MPR flooding.

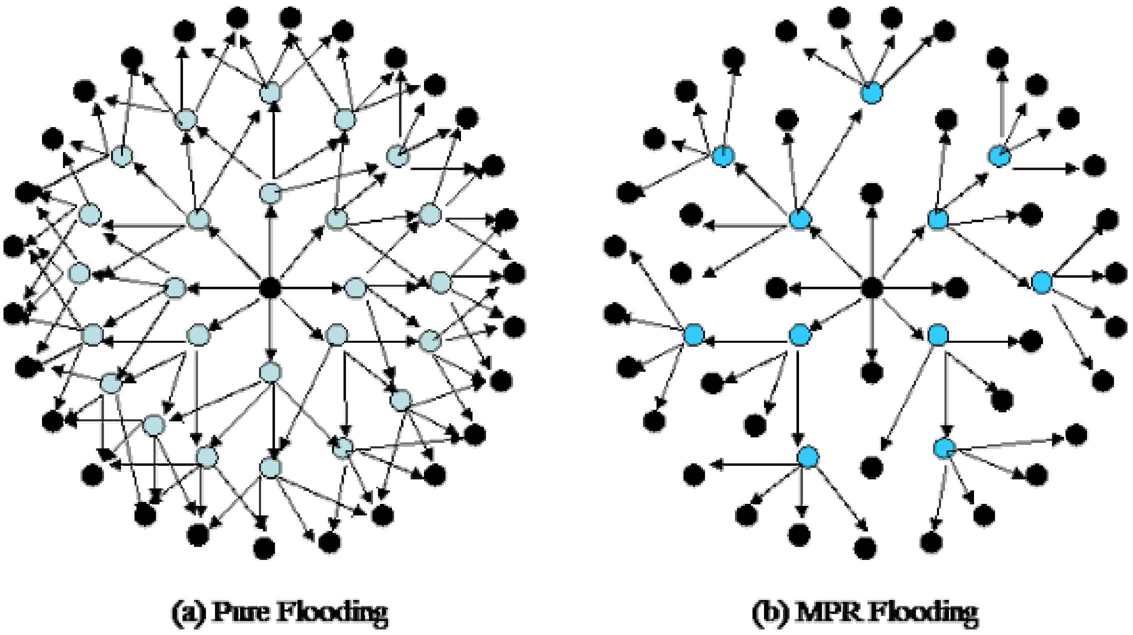


Figure 7. Comparing of pure flooding and MPR flooding types

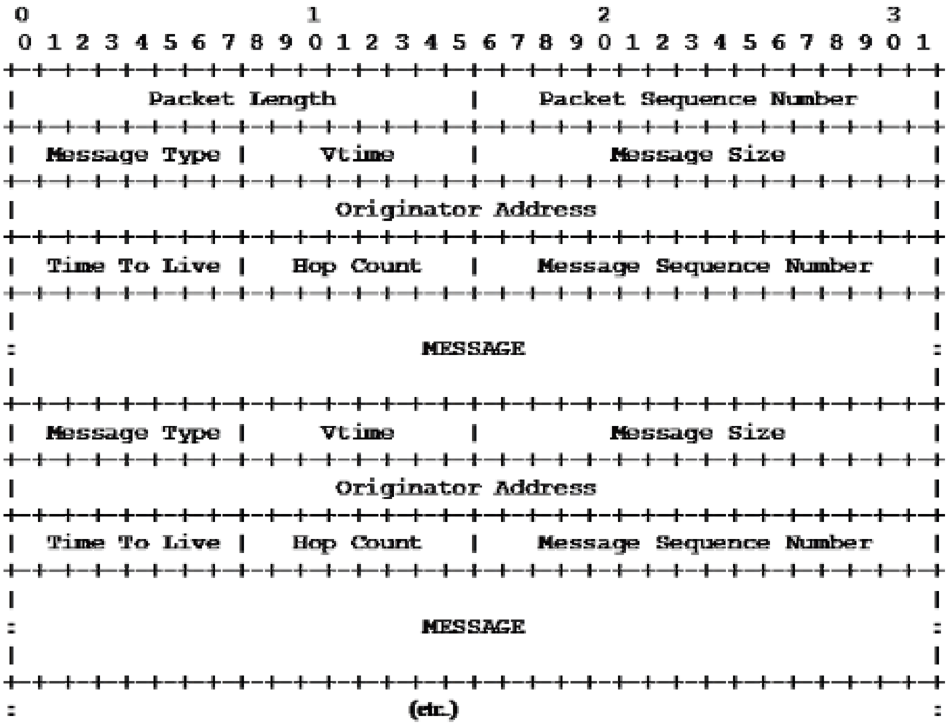
In fact, looking at Figure 7, it is possible to conclude that MPR nodes (blue nodes in (b)) form the routing "backbones" for the entire network and non-MPR nodes are somewhat like clients attached to MPRs. It is clear that in using MPR, OLSR has effectively reduced the routing overhead vis-à-vis flooding.

C. OLSR PACKET FORMAT

The fields in the OLSR packet header [OLSR 2003] are:

- Packet Length - length in bytes of the entire packet, including the header.
- Packet Sequence Number - A sequence number incremented by one each time a new OLSR message is transmitted by this host. A separate Packet Sequence Number is maintained for each interface so that packets transmitted over an interface are sequentially enumerated.

An OLSR packet body consists of one or more OLSR messages. Figure 8 shows a generic OLSR packet format [OLSR 2003].



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Packet Length         |    Packet Sequence Number     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Message Type |     Vtime     |         Message Size          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Originator Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time To Live |   Hop Count   |    Message Sequence Number    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:                            MESSAGE                            :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Message Type |     Vtime     |         Message Size          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Originator Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time To Live |   Hop Count   |    Message Sequence Number    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:                            MESSAGE                            :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                            (etc.)                             :
```

Figure 8. OLSR generic packet format (From: [OLSR 2003])

All OLSR messages must respect this header. The fields in the header are:

- Message type - An integer identifying the type of this message. Message types of 0-127 are reserved by OLSR while the 128-255 space is considered ``private" and can be used for custom extensions of the protocol.
- Vtime - This field indicates for how long after reception a node will consider the information contained in the message as valid.
- Message Size - The size of this message, including message header, counted in bytes.
- Originator Address - Source address of the originator of this message.
- Time To Live - The maximum number of hops this message can be forwarded. The use of this field can control the *radius* of flooding.
- Hop Count - The number of times the message has been forwarded.
- Message Sequence Number - A sequence number incremented by one each time a new OLSR packet is transmitted by this host.

D. DEFAULT VALUES FOR OLSR PARAMETERS

Certain default values for OLSR parameters have been suggested in section 18.2 and 18.3 of RFC 3626 [RFC 3626]. For *Hello intervals* and *Topology Control intervals*, they are 2 and 5 sec, respectively. The neighbor hold time(expiry time cache in the node) as well as the topology control hold time (expiry time cache in the MPR node) are respectively *three* times that of the Hello and Topology Control intervals. An attempt will be made to investigate the impact on ad hoc network performance by varying the Hello interval and the Topology Control interval in Chapter V.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. SIMULATION

A. INTRODUCTION

The simulation software used in this thesis is the *network simulator*, NS2 [NS2]. The software version used is the latest release at the time of the commencement of simulation, namely, ns-2.27, which can be downloaded from [NS2]. In fact, all previous versions prior to 2.27 are available at the same site for download. To complete the NS2 installation, it is possible to opt for the all-in-one version or download each component separately and compile them into a common directory. For ease of installation, the *all-in-one* package of version 2.27 has been chosen. NS2 has been chosen due to its availability. It is freely distributed and an open source. It does not consume an excessive amount of memory and a Pentium III computer with 128MB RAM has more than enough capacity to execute and run multiple simulations. In addition, many existing ad hoc routing protocols modules have already been implemented in NS2. Four such protocols are AODV, DSR, DSDV and TORA. However, OLSR was not implemented in NS2. It is necessary to acquire a compatible version of OLSR from the US Naval Research Laboratory website [NRL] and install the necessary modules so that NS2 can use OLSR protocol for network simulation. Many academics in their research in mobile ad hoc networking have widely accepted and used NS2. Thus, any simulations done using NS2 can be compared and referenced through the large number of examples available.

NS2 is a discrete-event driven simulation software targeted for network simulation. This software is currently maintained by the Information Science Institute of University of Southern California. Other network simulation tools include OPNET [OPNET], Glomosim [GLO], Qualnet [QUAL] and OMNET++ [OMN].

B. NETWORK SIMULATOR 2 (NS2)

1. Usage Process

The aim of this simulation tool is to allow researchers to study the extent of protocol interactions in the context of current and future network protocols. The bulk of the simulation tool is written in the C++ programming language and the Object Tool Command Language (OTCL). To write a simulation script, the user must use OTCL to define wireless mobile nodes in an enclosed network, the amount of traffic that is flowing, and which routing protocol is used. In addition, it is necessary to trace the mobility model used as well as the type of traffic at which level: routing, MAC or application. There are usually two types of output files: a trace file and a network animator (NAM) file. Trace files contains the events traces that can be further processed to understand the performance of the network. A NAM file allows the user to visually appreciate the movement as well as the interactions of the mobile nodes. Appendix A shows an example of an OTCL.

Figure 9 depicts the overall process of how a network simulation is conducted under NS2. Output files such as trace files have to be parsed to extract useful information. The parsing can be done using the *awk* command (in UNIX and LINUX, it is necessary to use gawk for the windows environment) or *perl* scripts. The results can be analyzed using Excel or Matlab to plot graphs. There are some software programs which can shorten the process of parsing trace files. [Tracegraph] has developed one such software program. However, it does not work well when the trace file is too large.

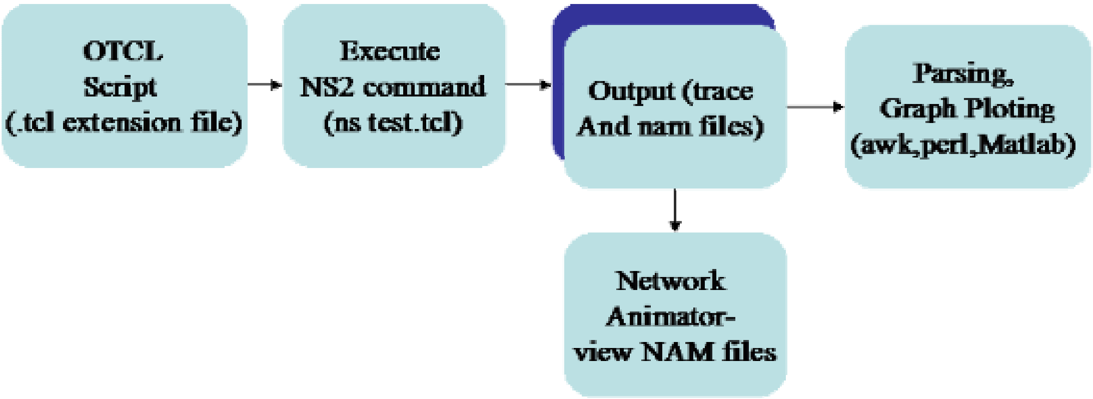


Figure 9. NS2 Simulation Process

2. Operating System and Memory

NS2 can be installed either in a UNIX (or LINUX) or Windows (2000 and XP) environment. However, in a Windows environment, it is necessary to install a Unix emulator such as Cygwin prior to the installation of the NS2 software. One disadvantage of performing the simulation in a Windows environment is the stability and support of the software. The simulations conducted for this thesis were all done in a Red Hat 9 LINUX environment. The software is highly stable in a LINUX environment. A fair amount of hard disk space (approximately 20 GB) must be allocated for this simulation purpose. Depending on the scale of simulation, for example, an ad hoc simulation of 50 nodes over 200s using OLSR protocol can generate up to 50 MB of trace and NAM files separately.

C. MOBILE NODE MODEL

Mobile nodes in NS2 make use of a routing agent to calculate routes from source to destination. In NS2, mobile nodes are implemented in the *MobileNode* class, which is derived from the parent class *node*. MobileNode has with added functionalities like movement and the ability to transmit and receive on a channel that allows it to be used to create mobile, wireless simulation environments. The mobility features, including node movement, periodic position updates, and maintaining topology boundary are implemented in C++. However, other network components within MobileNode itself (like classifiers, dmux , LL, Mac, and Channel) have been implemented in OTCL.

A mobile node is implemented with multiple components such as the application attached to it, a routing agent, link layer, MAC layer, and a queue. To complete this model, channel and propagation modeling are necessary to simulate the physical and wireless nature of radio communication. Figure 10 shows the model node [DOCU].

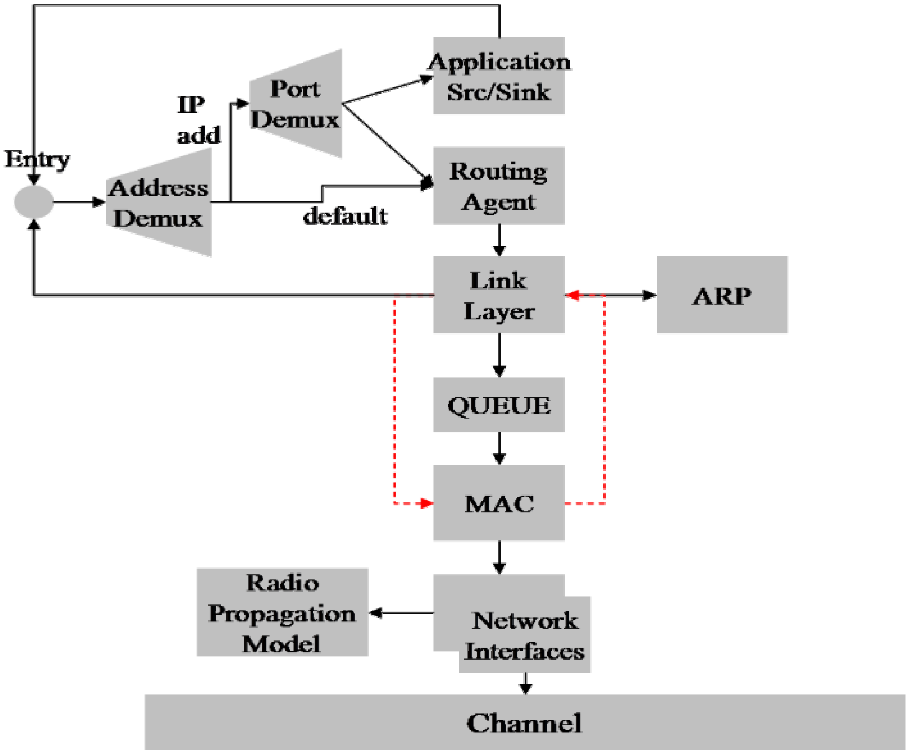


Figure 10. Mobile node (From: [DOCU])

An application such as TCP source packets or constant bit rate (CBR) packets is bound to a particular node and together with the routing agent, a path is determined to direct the data packet to its destination. This packet is passed onto the link layer, which also uses address resolution protocol (ARP) to obtain the neighbors' physical addresses, i.e., the MAC address. The packet is then queued until a positive signal is received from the MAC layer for transmission to the channel. Upon successful RTS/CTS signals at the MAC layer, the packet is delivered into the network interface. The packet is then duplicated and sent to all the network interfaces. Each network interface will provide the packet with receiving network interface information and then the propagation model is called upon.

1. 802.11 MAC Protocol

In NS2, there are two MAC layer protocols implemented for mobile networks: 802.11 and TDMA.

2. Radio Propagation Model

The radio propagation model uses Friss-space attenuation at near distances and an approximation to Two ray Ground at far distances, which assumes specular reflection off a flat ground plane.

3. Antenna

An omni-directional antenna having unity gain is used by mobilenodes.

4. Network Interfaces

The Network Interphase layer [reference NS2 document] serves as a hardware interface, which is used by a mobilenode to access the channel. The wireless shared media interface is implemented as a sub-class WirelessPhy (wireless physical medium) of the Phy (general physical layer) Class. The interface stamps each transmitted packet with information related to the transmitting interface such as the transmission power and wavelength. This is used by the propagation model in receiving network interface to determine if the packet has minimum power to be received and/or captured and/or detected (carrier sense) by the receiving node. The model approximates the Direct Sequence Spread Spectrum radio interface (LucentWaveLan direct-sequence spread-spectrum).

D. MOBILITY MODELS

Before describing the manner in which scenarios generated, it is necessary to understand the existing different mobility models [Camp 2002] used for the purposes of simulations. Three mobility models have been chosen and they are used in the simulation.

1. Random Waypoint Mobility Model

In the early days of network simulation studies, the Random Waypoint model [Maltz 1996] [Camp 2002] has been used extensively [Broch 1998], to evaluate the ad hoc routing protocols. Each host is initially placed at a random position within the simulation area. As the simulation progresses, each host pauses at its current location for a determinable period called the *pause time*. Pause time is used to overcome abrupt stopping and starting in the random walk model. Upon expiry of this pause time, the node will arbitrary select a new location to move towards it at a randomly selected velocity between a minimum and maximum value, which are stated at the start of the generation. Every host will continue this type of behavior throughout the entire duration of the simulation. Using this model, the hosts appear to move randomly within a confined compound.

The random waypoint model is selected for its simplicity. This simplistic modeling should be sufficient to capture the essence of the human mobility to make protocol evaluation academically meaningful. Taking a snapshot of a random number of people and observing their movement patterns in a chosen city area can make it possible to observe a certain state of *randomness* in their movement patterns.

2. Manhattan Grid Mobility Model

The deficiency of the random waypoint model is clearly in its unrealistic modeling of real life activity. When people move from one point to another, they are somewhat driven by objectives and physical constraints within an environment. For example, it is necessary to walk *around* buildings and not *through* buildings. As such in urban landscapes, a random waypoint may be grossly ineffective in capturing the real movements of people. The Manhattan Grid model [Tech 1998] [Camp 2002] is proposed for the urban setup. A city is usually formed from “grids”, which are actually areas formed by intersecting lines running parallel and horizontal to each other. The size of the grid indicates, to a certain extent, the degree of urbanization of the city. A large city has small grids while some have larger ones. Although this model is not very accurate as far as older cities such as London and Paris are concerned. The grids in these cities are not necessarily formed by ninety-degree intersecting roads but a huge variety of roads intersecting, forming many interesting shapes and sizes. Nodes will move along the grids, and for the purpose of this simulation, they are confined to four directions: left, right, up and down.

The Manhattan model can be described by the following parameters: mean speed, minimum speed (with a defined standard deviation for speed - normal distribution), a probability to change speed at position update, and a probability to turn at cross junctions.

The generated mobility file was modified to reflect the military scenario more correctly. In urban warfare, tanks and troops do not move in haphazard fashions because they want to avoid crossing each other's weapon line of sight. They are oriented in specific positions and move coherently towards a specific target. In the simulation for this thesis of the Manhattan Grid model, such a setup was chosen. Initially, nodes are positioned randomly surrounding a target reference. Over time, each node will move in a grid-like fashion but gravitate towards the same common objective. The nodes will zoom into the target of interest. In this case, the centre of the simulation area was chosen as the target of interest. When nodes move towards the target, they are still governed by the parameters previously described. See Figure 11 and Figure 12 for the start and end of simulation diagrams.

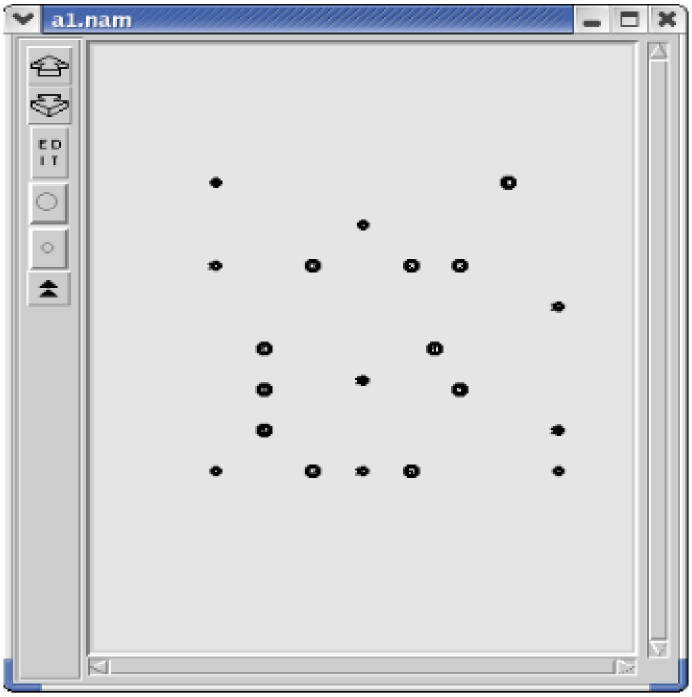


Figure 11. Nodes Setup at beginning of simulation

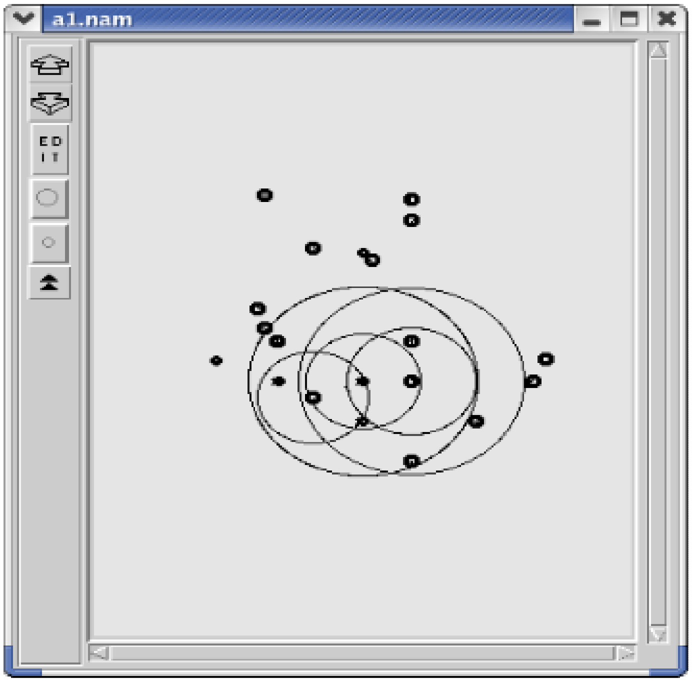


Figure 12. Nodes final position at end of simulation

3. Reference Point Group Mobility (RPGM) Model Generation

In the RPGM model [Hong 1999] [Camp 2002], nodes *cluster* together to form groups. Together they move towards a specific target in unison as a group. Group nodes are bounded within a certain distance between each other. The logical centre (i.e., the reference point) of this group defines its movement. Within a group, each node has its own movement vector, which is still confining the node within the vicinity defined by the *radius* of the logical centre. Group motions are often generated as a series of random movements forming a path. Such a path is given by defining a sequence of check points along the path corresponding to given time intervals. Nodes movements within a group can also be random within the group. Each time the group reaches its destination, they pause for some time before moving on. This model is parameterized by the following factors:

- Number of nodes per group
- Group change probability – whereby nodes *migrate* from one group to another
- Maximum distance to center of group
- Group size standard deviation

By proper selection of these check points, it is easily possible to model many realistic situations, such as reaching pre-defined destinations within a given time limit. The RPGM general framework model allows for high flexibility in the design of specific mobility models. See Figure 13 and Figure 14 for the positioning of nodes before and after simulation.

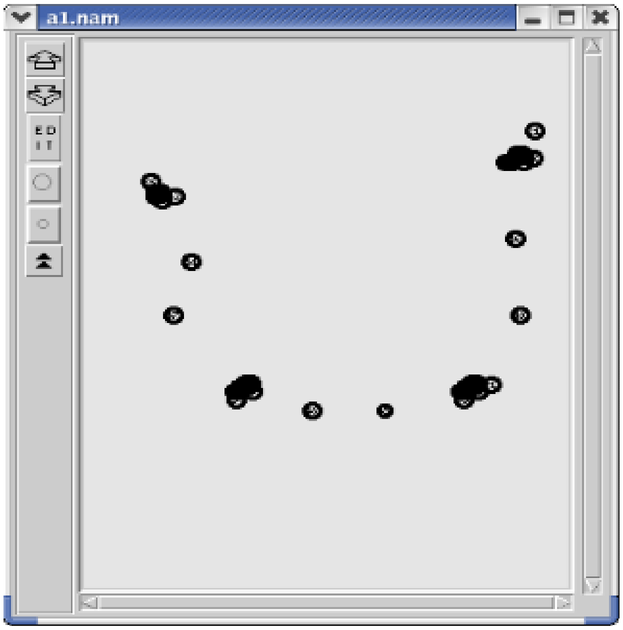


Figure 13. Nodes Setup at beginning of simulation

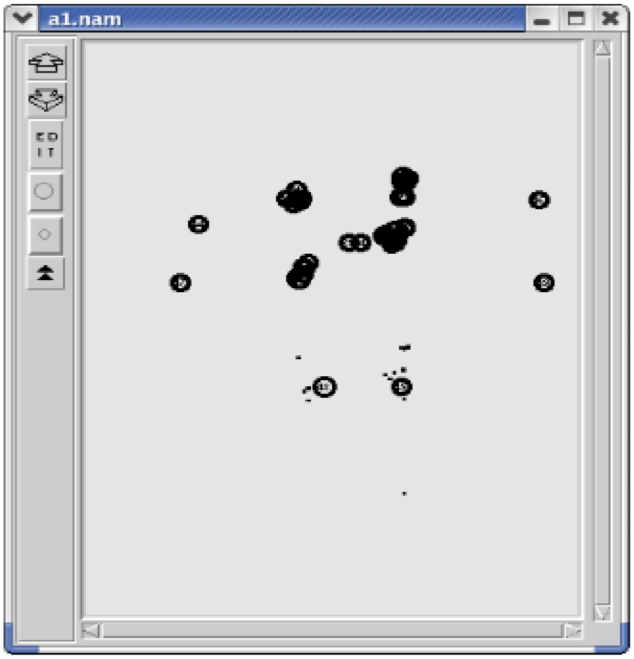


Figure 14. Nodes final position at end of simulation

RPGM models have the potential to be the closest mobility model for a military operation environment because military troops move in groups such as battalions or divisions. Tanks, troops, ships etc tend to move in groups rather than individual elements. RPGM has the advantage of defining the group movement for military groups. To cater to realistic military operation, the generated RPGM models was modified. Groups are separated in a ring-like fashion and moving towards an objective, similar to the case of the Manhattan Grid model. The simulation is a simplistic target attacking scenario, although more sophisticated movements can be made in the future to cater to different possibilities. Since groups are created and nodes are all allocated to the specific groups, there could be no connectivity between the nodes at the beginning of the simulation due to the far distance separating them. However, in real life, the military deploys signal relay nodes such as mobile communication trucks or even mobile satellites in fixed positions, to bridge the communication gaps of these groups, which were modeled into the simulation setups as well.

E. TRAFFIC GENERATION

Two types of traffic can be generated for the purpose of simulation: constant bit rate (CBR) traffic or transmission control protocol (TCP) traffic. All simulations used

CBR traffic type as the source of data traffic. CBR presents a more stringent demand on the mobile ad hoc network. CBR and TCP (in fact, it is a FTP application) traffic can be generated using pre-built in OTCL scripts (cbrgen.tcl) in the NS2 directory. Traffic patterns can be generated using the cbrgen.tcl script and it has few parameters to input. An example of execution of the script is:

```
ns cbrgen.tcl –type cbr –nn 20 –seed 1.0 –mc 10 –rate 10.0 > cbr-20-c10r10
```

with parameters,

-*type* either CBR or TCP traffic

-*nn* which is the number of node(s) to be simulated

-*seed* is the seed to the random number generator

-*mc* which is the maximum number of connections (pair-wise)

-*rate* which is the rate at which one source generates traffic in packets/second

The output of the traffic generated is stored in the file *cbr-20-c10r10*. An example of the file content is shown in Appendix B. From the content of the generated traffic file, connection loading is generated to commence at a specific time and will last throughout the simulation run. The purpose is to stretch the network to its limit and observe the performance when the routing protocols are put to the stress test. The time at which nodes start to transmit is also arbitrarily selected by the algorithm. The *interval* of packet generation is the actual load since a shorter interval means that more traffic packets are generated within a fixed period of time. Random seeds can be specified to improve the scripts' *randomness*.

F. SCENARIO GENERATION

1 Random Waypoint Model Generation

NS2 can generate random waypoint mobility using a function that comes with the installation software – *setdest*. Setdest is located in the sub folder ../ns-2.27/indep-utils/cmu-scen-gen/setdest/ directory. Two versions of setdest are available. In version 1, the command as well as the parameters available is:

```
./setdest –v 1 –n 20 –p 15.0 –M 20.0 –t 200.0 –x 800 –y 800
```

where

-*v* is the version of the setdest function.

-*n* is the number of nodes under simulation

-*p* is the pause time in seconds

-*M* is the maximum speed allowable

-*t* is the total simulation time

-*x* is the length of the simulation space, assuming two-dimensional

-*y* is the breadth of the simulation space, assuming two-dimensional

In version 2, the format and parameters are :

```
./setdest –v 2 –n 20 –s 1 –m 2.0 –M 10.0 –t 200.0 –P 1 –p 10.0  –x 800 –y 800
```

where

-*v* is the version of the setdest, which is 2 here

-*n* is number of nodes

-*s* is the speed selection type, 1 for uniform distribution between minimum and maximum speed. 2 is for normal distribution between the minimum and maximum speed.

-*m* is the minimum speed

-*M* is the maximum speed

-*t* is simulation time

-*P* is the pause selection type, 1 for constant and 2 for uniform distribution of [0,2x pause time]

-*p* is the pause time

-*x* is the length of the simulation space, assuming two-dimensional

-*y* is the breadth of the simulation space, assuming two-dimensional

2. Manhattan Grid Model Generation

The BonnMotion software developed by [Waal 2003] was used to generate the Manhattan Grid models. It is a software program developed in Java, which creates and analyses mobility scenarios. BonnMotion has been developed within the Communication Systems group at the Institute of Computer Science IV of the University of Bonn. The mobility movements scripts created can be ported over to NS2 as well as other network simulation tools such as QualNet.

3. Reference Point Group Mobility Model Generation

Likewise, the BonnMotion software can also be used to generate RPGM models. These models must be converted to NS2 mobility script files in order to be integrated into the OTCL scripts for simulation. Note that the BonnMotion software, which is written in Java, can be installed and executed in any OS platform. In this thesis, all the simulation and mobility scripts were generated in the LINUX OS. See Appendix C for a sample mobility script generated by the BonnMotion software.

G. OLSR INSTALLATION

In the default setup of NS2, only the AODV, DSR and DSDV implementations were installed. OLSR does not come with the NS2 version 2.27. A working version was downloaded from the US Naval Research Laboratory website [NRL], which is compatible with the NS2 version,

In the installation process of NRLOLSR, it is necessary to make changes to the C++ and header files in the NS2 directory. Recompilation of the entire NS2 must be done whenever changes are made to the C++ or header files. This can be done by executing a *make* command at the NS2 directory.

There is, however, a bug that is not highlighted during the installation process. The file nsProtoSimAgent.cpp as a minor error in the compilation process. The type class was declared wrongly at line 100 :

```
char* nodeName = tcl.result();
```

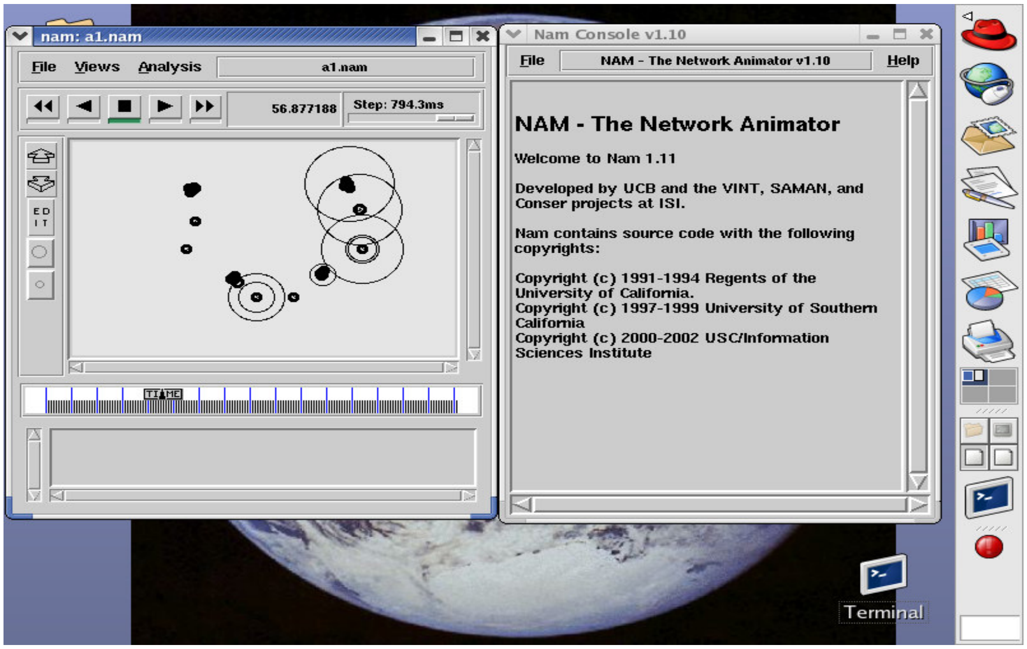
should read

```
const char* nodeName = tcl.result();
```

Upon successful installation, make an attempt to execute the sample OLSR script file located in ns-2.27/nrlolsr/ns directory and compare the results of this file execution to the standard results expected. This ensures that NRL OLSR has been installed properly and is functioning correctly.

H. DATA TREATMENT

As mentioned earlier, two files can be generated at the end of a simulation run: an event trace file and a network animation NAM file. The trace events can be logged at the application level (CBR, TCP traffic agent), routing layer, MAC layer as well as the node's movements at specific intervals. The NAM files provide a visual appreciation of the entire node's movement and interaction with other nodes and thus enables the user to obtain a visual validation of the simulation model. Both are CPU-intensive jobs and consume quite a significant amount of memory. A typical trace file with all events logging turned on can generate up to 20 to 50 MB of data for each 200s simulation run. This value will vary from one routing protocol to another. A typical trace file can be seen in Appendix D, but only partial information is displayed. Figure 15 shows the NAM application and a sample NAM graphic file.



NAM - The Network Animator

Welcome to Nam 1.11

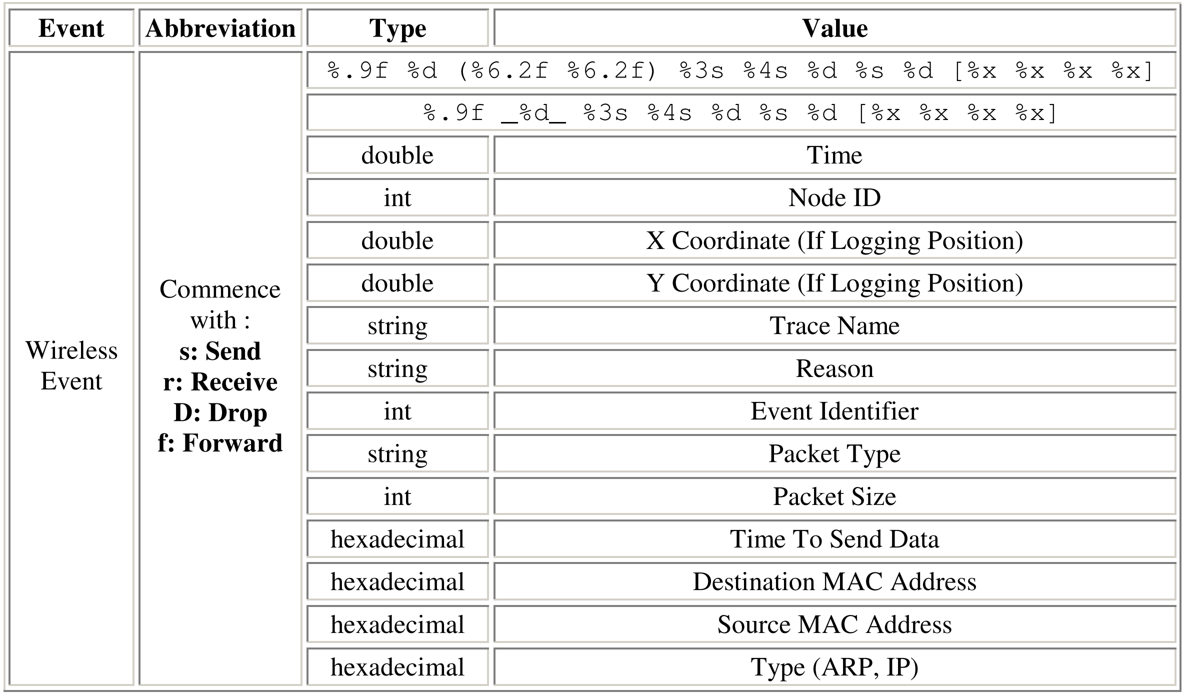
Developed by UCB and the VINT, SAMAN, and Conser projects at ISI.

Nam contains source code with the following copyrights:

Copyright (c) 1991-1994 Regents of the University of California.
Copyright (c) 1997-1999 University of Southern California
Copyright (c) 2000-2002 USC/Information Sciences Institute

Figure 15. NAM Application in Linux OS

There exist two formats for trace files (old and new). In the old trace format for wireless simulation traces, a trace file [TRACE] begins with either character (s, r, D or f) that can be found in Table 4.



Event	Abbreviation	Type	Value
Wireless Event	Commence with : **s: Send** **r: Receive** **D: Drop** **f: Forward**	%.9f %d (%6.2f %6.2f) %3s %4s %d %s %d [%x %x %x %x]	
		%.9f _%d_ %3s %4s %d %s %d [%x %x %x %x]	
		double	Time
		int	Node ID
		double	X Coordinate (If Logging Position)
		double	Y Coordinate (If Logging Position)
		string	Trace Name
		string	Reason
		int	Event Identifier
		string	Packet Type
		int	Packet Size
		hexadecimal	Time To Send Data
		hexadecimal	Destination MAC Address
		hexadecimal	Source MAC Address
		hexadecimal	Type (ARP, IP)

Table 4. Wireless event trace file (From: [TRACE])

Each field that succeeds the first character is separated by an empty space. Table 5 shows the significance and meaning of each field.

The new format for wireless simulation has similar starting action flags. See Table 6 [TRACE]. Each parameter that follows suit begins with a dash and some letters stating its type.

The first letter of flags with two letters designates the flag type:

- N: Node Property
- I: IP Level Packet Information
- H: Next Hop Information
- M: MAC Level Packet Information
- P: Packet Specific Information

To extract information from the generated trace file, it is possible to an awk or perl script . Appendix E contains examples of the awk and perl scripts. This awk script extracts the packet delivery ratio, delay and routing overheads. The perl script is used to generate repetitive simulation runs. Awk and perl functions are already pre-installed in REDHAT LINUX OS 9. For the windows environment, it is possible to download gawk and perl software from [AWK PERL].

Event	**Type**	**Value**
ARP Trace	`------- [%s %d/%d %d/%d]`	
	string	Request or Reply
	int	Source MAC Address
	int	Source Address
	int	Destination MAC Address
	int	Destination Address
AODV Trace	`[0x%x %d %d [%d %d] [%d %d]] (REQUEST)`	
	hexadecimal	Type
	int	Hop Count
	int	Broadcast ID
	int	Destination
	int	Destination Sequence Number
	int	Source
	int	Source Sequence Number
	`[0x%x %d [%d %d] %f] (%s)`	
	hexadecimal	Type
	int	Hop Count
	int	Destination
	int	Destination Sequence Number
	double	Lifetime
	string	Operation (REPLY, ERROR, HELLO)
IP Trace	`------- [%d:%d %d:%d %d %d]`	
	int	Source IP Address
	int	Source Port Number
	int	Destination IP Address
	int	Destination Port Number
	int	TTL Value
	int	Next Hop Address, If Any
TCP Trace	`[%d %d] %d %d`	
	int	Sequence Number

Event	Type	Value
	int	Acknowledgment Number
	int	Number Of Times Packet Was Forwarded
	int	Optimal Number Of Forwards
CBR Trace	[%d] %d %d	
	int	Sequence Number
	int	Number Of Times Packet Was Forwarded
	int	Optimal Number Of Forwards

Table 5. Wireless packet trace format (From: [TRACE])

Event	Abbreviation	Flag	Type	Value
Wireless Event	s: Send r: Receive d: Drop f: Forward	-t	double	Time (* For Global Setting)
		-Ni	int	Node ID
		-Nx	double	Node X Coordinate
		-Ny	double	Node Y Coordinate
		-Nz	double	Node Z Coordinate
		-Ne	double	Node Energy Level
		-Nl	string	Network trace Level (AGT, RTR, MAC, etc.)
		-Nw	string	Drop Reason
		-Hs	int	Hop source node ID
		-Hd	int	Hop destination Node ID, -1, -2
		-Ma	hexadecimal	Duration
		-Ms	hexadecimal	Source Ethernet Address
		-Md	hexadecimal	Destination Ethernet Address
		-Mt	hexadecimal	Ethernet Type
		-P	string	Packet Type (arp, dsr, imep, tora, etc.)
		-Pn	string	Packet Type (cbr, tcp)

Table 6. New Trace format for wireless packets (From: [TRACE])

I. PERFORMANCE METRICS

1. Packet Delivery Ratio Calculation

The packet delivery ratio is expressed as the percentage of CBR data traffic that has been received by all destinations (sinks) over the total number of packets sent by all the sources within the period of simulation.

$$\text{Packet Delivery Ratio (PDR)} = \frac{\sum \text{CBRpackets received by all sinks}}{\sum \text{CBR packets sent by all sources}}$$

The packet delivery ratio can be interpreted as the loss ratio that will be experienced at the routing layer which in turn has an impact on the overall throughput of what the network can support. It is a fundamental characterization of the performance of routing protocols.

2. Network Delay Calculation

Network delay is defined as the average delay experienced by all connections throughout the simulation experiment. Each data transmission between a source and a destination will experience a network delay in the network. The delay is defined as the difference in time the moment all transmission of packets are delivered and the time these packets are all actually received. The aggregate of all such connections delays is found and averaged over the total number of transmission connections pairs. Delay is a significant factor due to the necessity to provide low latency applications such as video-on-demand and other time-sensitive applications, e.g., IP telephony. It typifies the suitability of using certain routing protocols to support these applications.

$$\text{Network Delay} = \frac{\sum \{\text{Time}_{\text{data received at destination}} - \text{Time}_{\text{source want to send}}\}}{\text{\# of connection pairs}}$$

3. Routing Overhead Calculation

The routing overhead is the total amount of control data packets sent by the routing protocol throughout the duration of the simulation. Each time a packet is forwarded over multiple hops, routing overhead is counted as many packets as there are hops. The amount of routing overhead is a significant factor to determine the efficiency of the routing protocol. Highly efficient routing protocols have lower routing overheads so as to maintain faster route convergence, and thereby, lower overall delay. Such protocols whose routing overheads are low will enable the protocol to scale better and consume less energy. If more control packets are sent by routing agents, the chance of collision for the transmission channels increases, and thus causes the delay of the application to increase indirectly.

4. Energy Consumption

During the introduction of this thesis, we have highlighted the importance of power availability in a mobile ad hoc networking environment. Mobile nodes will most likely run on batteries and there will not be a constant, permanent source of power supply as in the case of fixed nodes. The energy consumption by the communication protocol at the routing layer is our primary concern. The energy model in a node has a initial value which is the level of energy the node has at the beginning of the simulation. For every packet the node transmits and receives, a certain amount of power is consumed. Transmit power consumes more power than receiving information and therefore is given a bigger value. When the energy level at the node drops to zero, no more packets can be received or transmitted by the node and the node is essentially turned off. We define the Total System Energy as the sum of energy levels of each of the nodes within the system. In some cases, we consider the Final System Energy state which is defined as the total energy of all the nodes combined at the final state when the simulation duration has ended. Mathematically, we express,

$$\text{Total System Energy at time t} = \sum_k \text{Energy of Node k at time t}$$

$$\text{Final System Energy} = \sum_k \text{Energy of Node k at end of simualtion}$$

V. SIMULATION – RESULTS

For all the simulation results reported in this thesis, each data point is the average over at least three simulation runs using the same set of parameters but different random seeds. This is to ensure that the data collected is average out and to prevent fluke results.

A. RANDOM WAYPOINT MOBILITY MODEL

1. Mobility - Results

In this simulation case, we consider a map size of 670m by 670m. Four routing protocols are simulated: AODV, OLSR, DSR, DSDV. The node speed varies from 5m/s (18 km/h - walking quickly) to 25m/s (90 km/h – fast vehicular speed), in increments of 5m/s. Each node has a pause time of 2s to simulate a high tempo military situation. The traffic type is CBR, the application agent is sending at a rate of 10 packets per second (data) continuously. The entire simulation run lasts for 200s, and consider transmission power consumption to be higher than the receiving power consumption rate. Each node is pre-determined with 10J of energy. Table 7 summarizes the simulation parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	670m x 670m
Speed	5-10-15-20-25m/s
Mobility Model	Random Waypoint
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10 pkts/sec
Pause Time	2s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	10

Table 7. Simulation Parameters for Random Waypoint – Mobility Variations

The packet delivery ratio and the routing overhead for this simulation are shown in Figure 15 and Figure 16, respectively. On-demand routing protocols consistently outperformed the table-driven protocols in terms of packet delivery rate regardless of the node speed. Among the four, the DSDV routing protocol has the worst results. For on-demand protocols, the packet delivery ratio is relatively constant and close to 100% delivery rate. Therefore, it is the most suitable consideration if data delivery is of the utmost consideration. Moreover, OLSR has a considerable amount of routing overheads compared to the other three and DSDV has the least routing overhead. The tradeoff for DSDV routing is that, what it lost out on the packet delivery ratio, is gained in terms of a shorter network delay. See Figure 17. AODV and DSR have their delay almost doubled when mobility increases five fold. DSR has the worst network delay performance amongst the four. For delay-sensitive applications, DSDV may be considered since its network delay remain relatively low and fluctuates little as speed increases. Also, another table-driven protocol, OLSR has relatively small network latency, and therefore, is a good compromise candidate for a relatively acceptable high packet delivery ratio and low network latency.

The system energy state for each simulation case is plotted against varying mobility speeds (from 5 to 25 m/s) in Figure 18 to Figure 22. The system energy is defined as the total sum of all the energies of each individual node at a particular state (time) of the simulation. As shown in these diagrams, DSDV consistently conserved more energy for the system and the rate of energy consumption is slower as simulation time increases. Despite having a high number of routing overheads, OLSR did not perform too badly and has consistently better results than AODV and DSR.

Overall, taking into consideration all the assumptions and the metrics that have been used in this thesis, OLSR has presented itself as a relatively more credible routing protocol than the other four. However, these results are not conclusive, as the mobility model chosen in this simulation experiment does not reflect real life situations very accurately, especially where military operations are concerned.

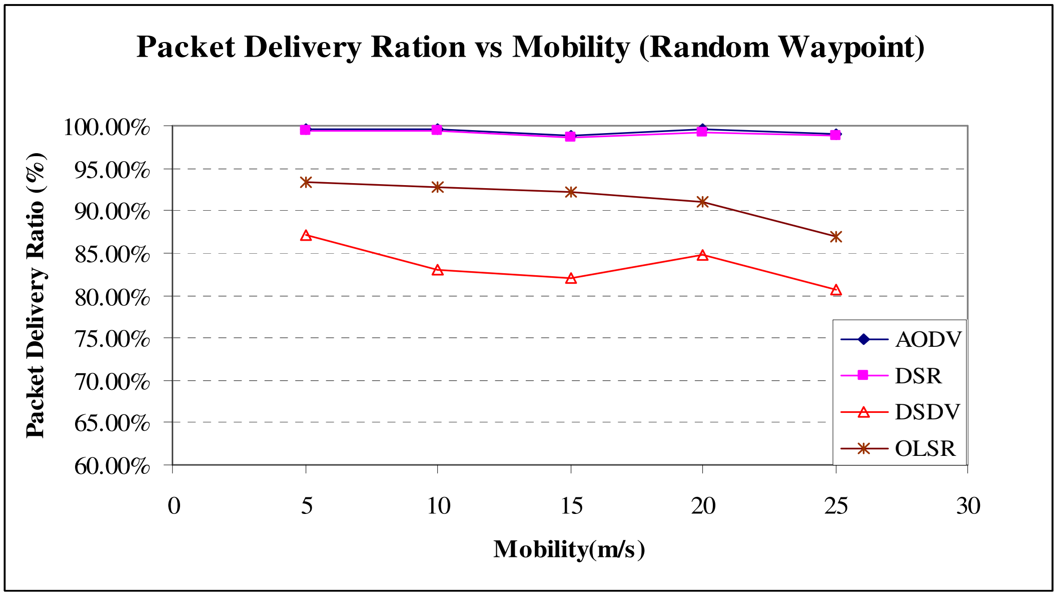


Figure 16. Packet Delivery Ratio with varied Speed (5 -25 m/s) – Random Waypoint

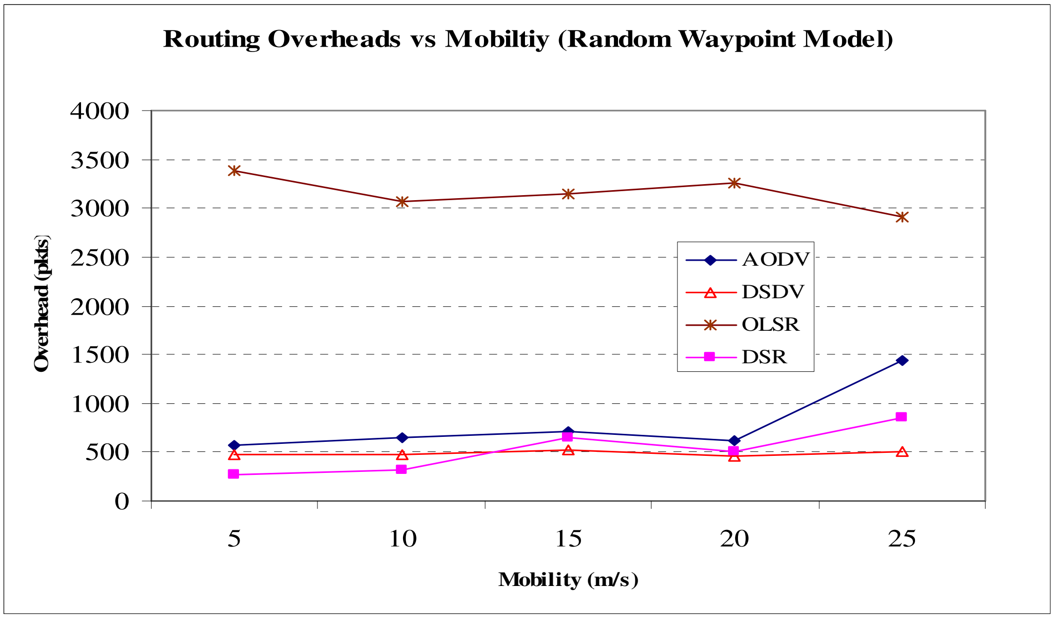


Figure 17. Routing Overhead with varied Speed (5 -25 m/s) – Random Waypoint

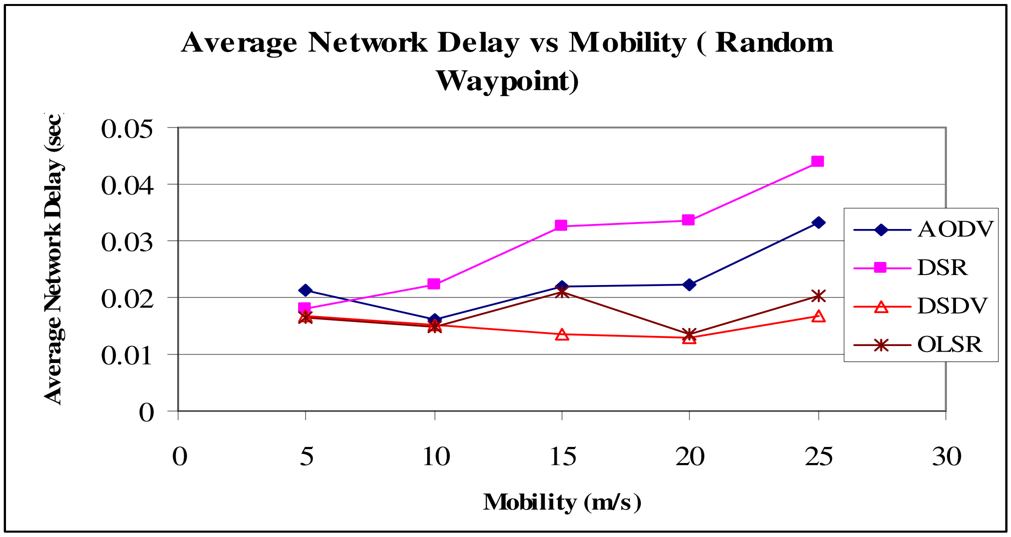


Figure 18. Average Network Delay with varied Speed (5 - 25 m/s) – Random Waypoint

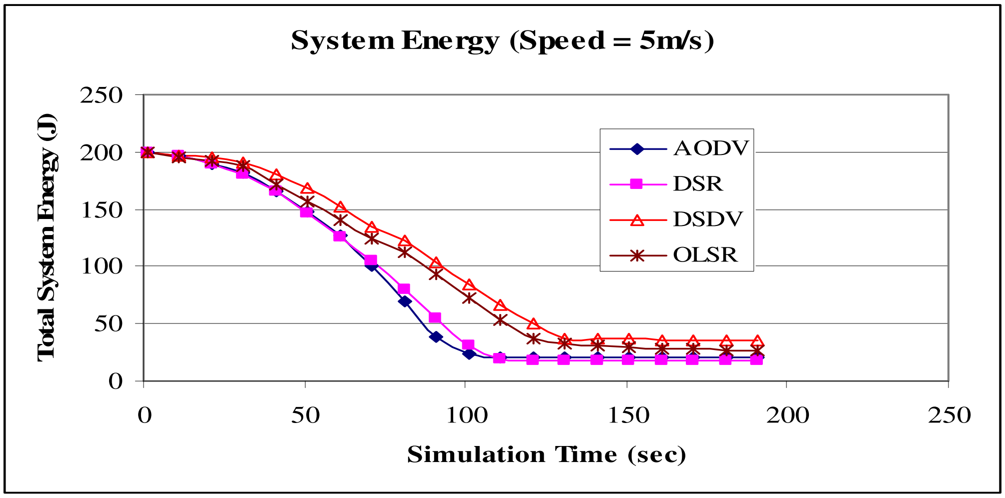


Figure 19. System Energy at Speed = 5m/s

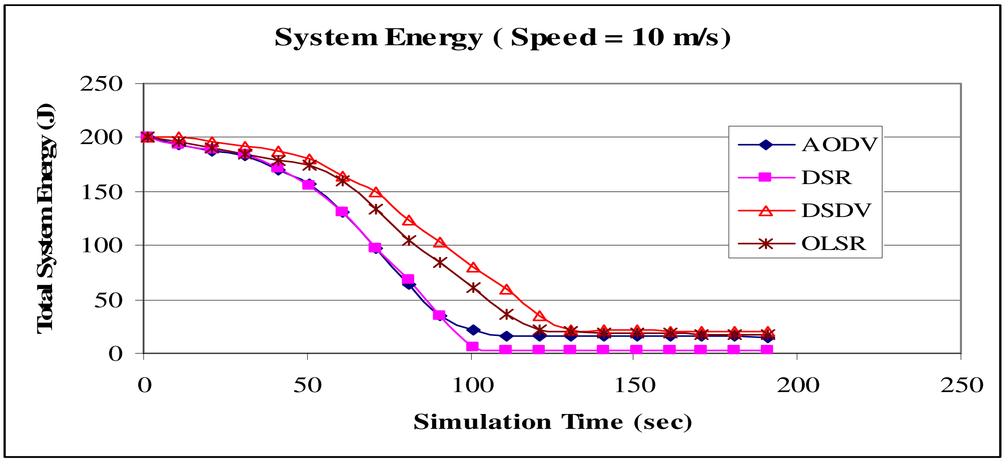


Figure 20. System Energy at Speed = 10m/s

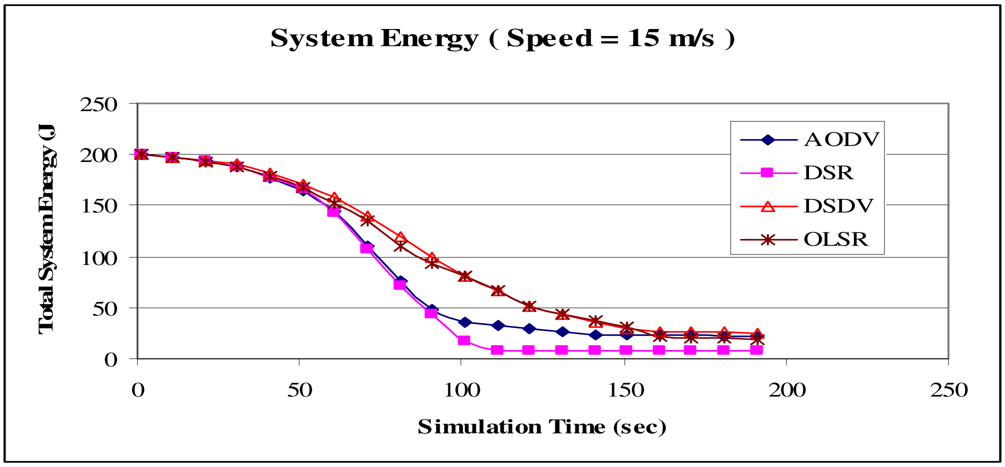


Figure 21. System Energy at Speed = 15 m/s

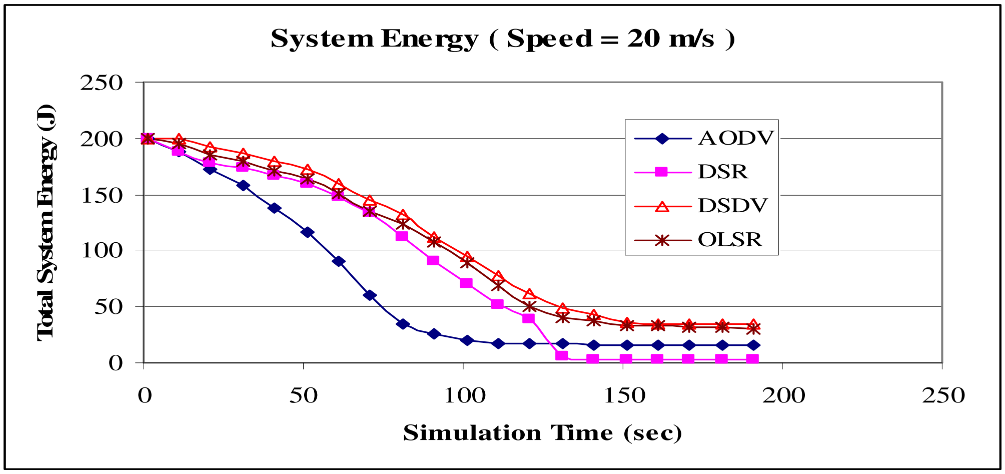


Figure 22. System Energy at Speed = 20 m/s

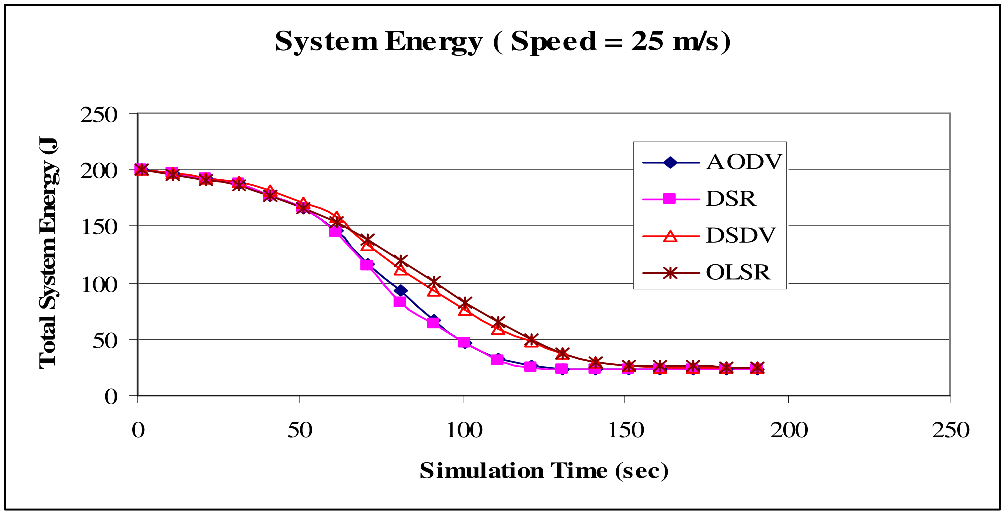


Figure 23. System Energy at Speed = 25m/s

2. Node Density - Results

In the second set of simulations, the node density of the simulation network was varied to ascertain the performance impact of the four routing protocols. The node density is increased from 10 to 50 nodes within the same map area as each simulation run is executed. The goal is to understand the impact of having more nodes within a fixed area of operation on the network's packet delivery ratio and its delay as well as the overall routing overheads. The traffic type chosen in this thesis, is again, CBR, and each node is sending at a rate of 10 pkts/sec. Again, the rest of the simulation parameters are relatively unchanged and the speed of the nodes was maintained at 10m/s. The entire simulation run lasts for 200s.

Table 8 summarizes the simulation parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	10-20-30-40-50
Map Size	670m x 670m
Speed	10 m/s
Mobility Model	Random Waypoint
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10 pkts/sec

Simulation Parameters	
Pause Time	2s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	10

Table 8. Simulation Parameters for Random Waypoint – Node Density Variations

Figure 23 shows the results of the simulation, which demonstrates the overall packet delivery ratio of the network using the routing protocols. DSR and AODV, in general, have better packet delivery performance than DSDV and OLSR. However, as the higher node density is approached, the performance output of OLSR and AODV does not differ greatly. The graphical results have shown that on-demand protocols have consistently performed better than table-driven protocols, in the case of varying node density.

Figures 24 and 25 shows the routing overheads as well as the network delay incurred. From the graphs, it is possible to see that OLSR has a significant amount of routing overheads compared to the rest of the routing protocols. The parameters used in the OLSR routing protocols such as the hello intervals and topology control intervals are set to be the default values as provided by the RFC 3626. The hello interval and topology control interval are set at 2.0 sec and 5.0 sec, respectively. For DSDV, the periodic update of routes is set default at 15 sec. In that respect, the expectation is that DSDV would have significantly less overheads being generated, nearly 3 to 7.5 times less. At the peak density of 50 nodes, OLSR has a seven fold increase in routing overheads than any of the remaining routing protocols. Although OLSR employs MPRs to reduce routing overheads (as compared to broadcast flooding), however, the randomness in node positioning and movement of nodes has resulted in the inefficient usage of the MPR flooding.

Figure 26 to Figure 30 highlight the total energy system change with respect to time. Again, as in the case of varying mobility, DSDV has performed better than all other routing protocols. It is interesting to note that, between the period of 20 to 100 sec, the difference in the system energy is the greatest. This can be attributed to a surge in routing activities as nodes begin to transmit and receive data. In general, on demand protocols consume more energy within the system at the start of the simulation until midway where table-driven protocols exhaust their nodes faster when the total system energy starts to get low.

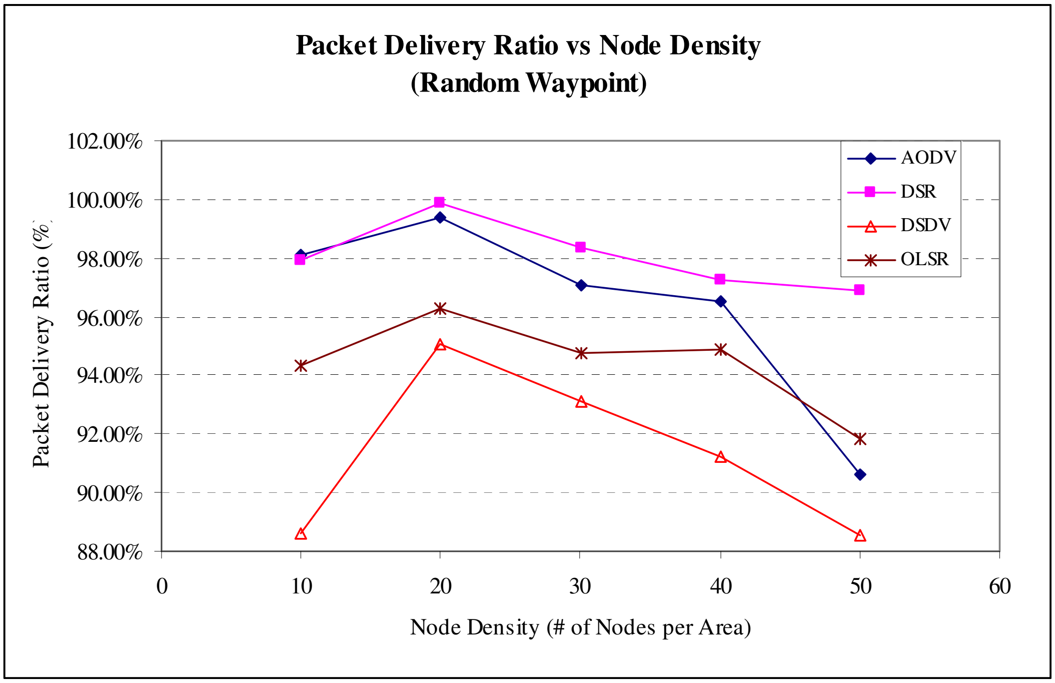


Figure 24. Packet Delivery Ratio with varied Node Density (10 - 50 nodes per area)

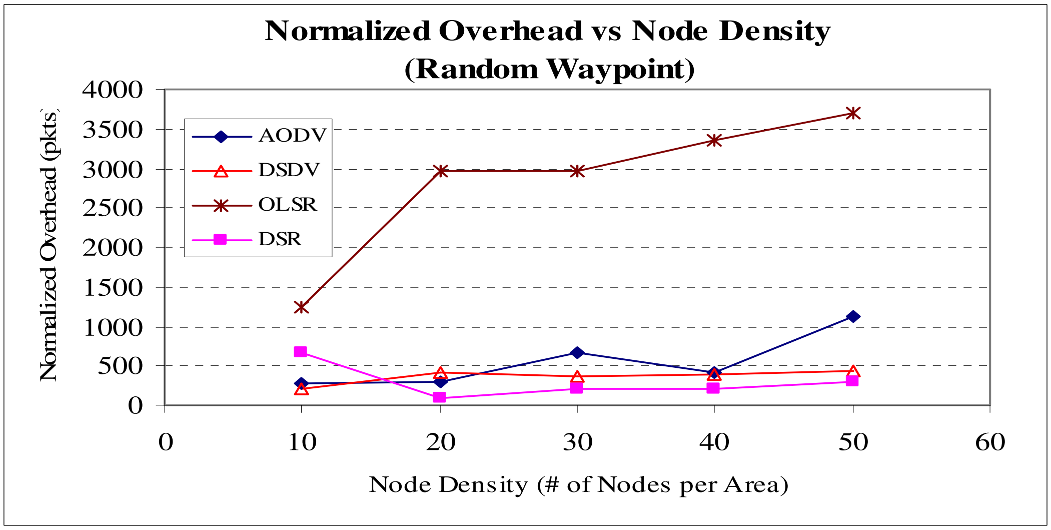


Figure 25. Normalized Routing Overheads with varied Node Density (10 - 50 nodes per area)

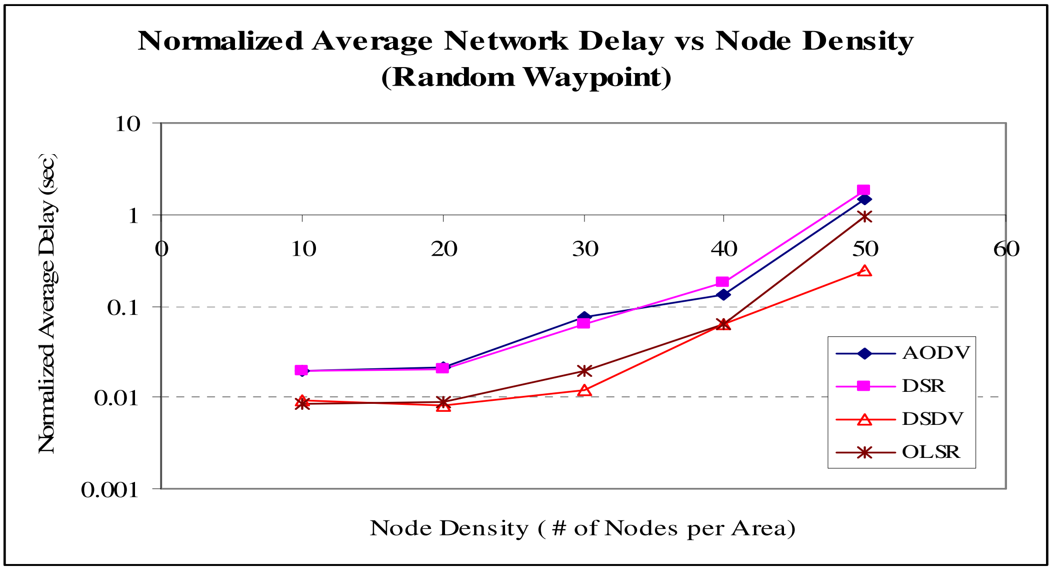


Figure 26. Normalized Average Delay with varied Node Density (10 - 50 nodes per area)

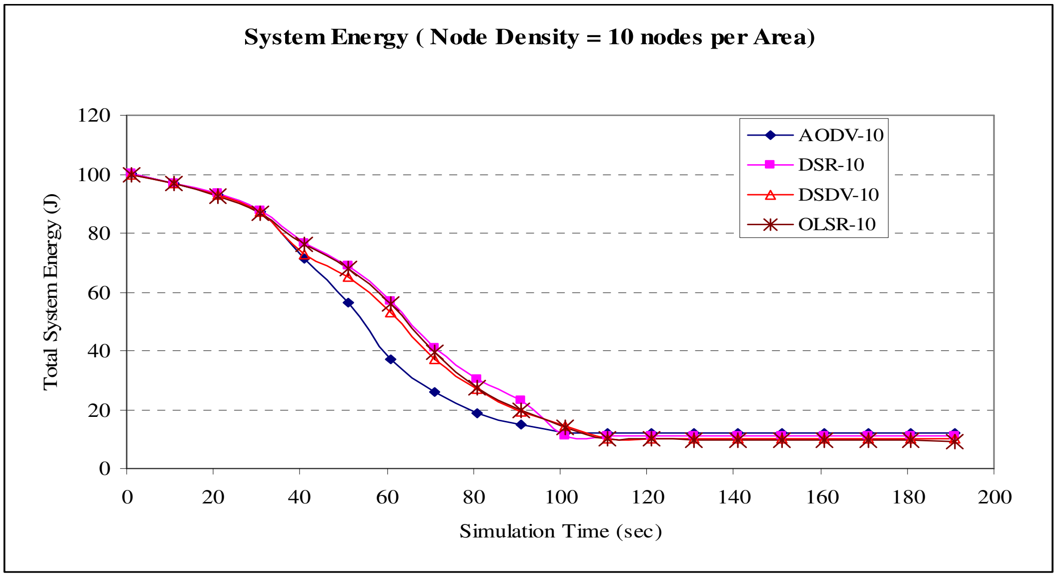


Figure 27. System Energy for Node Density = 10 Nodes per Area

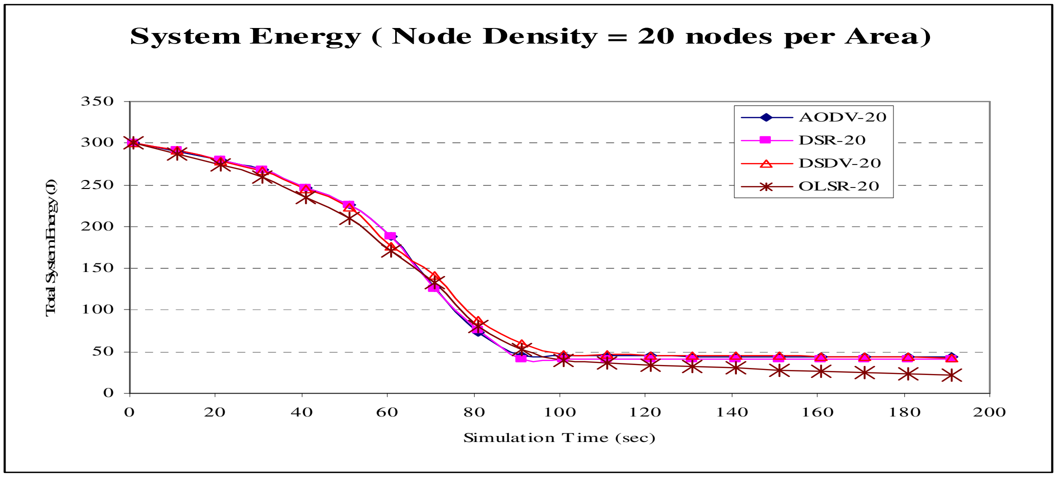


Figure 28. System Energy for Node Density = 20 Nodes per Area

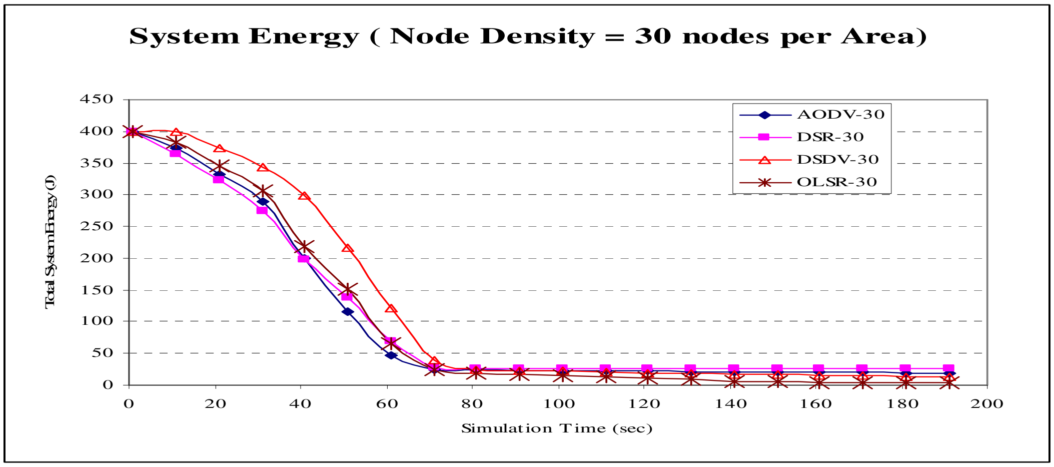


Figure 29. System Energy for Node Density = 30 Nodes per Area

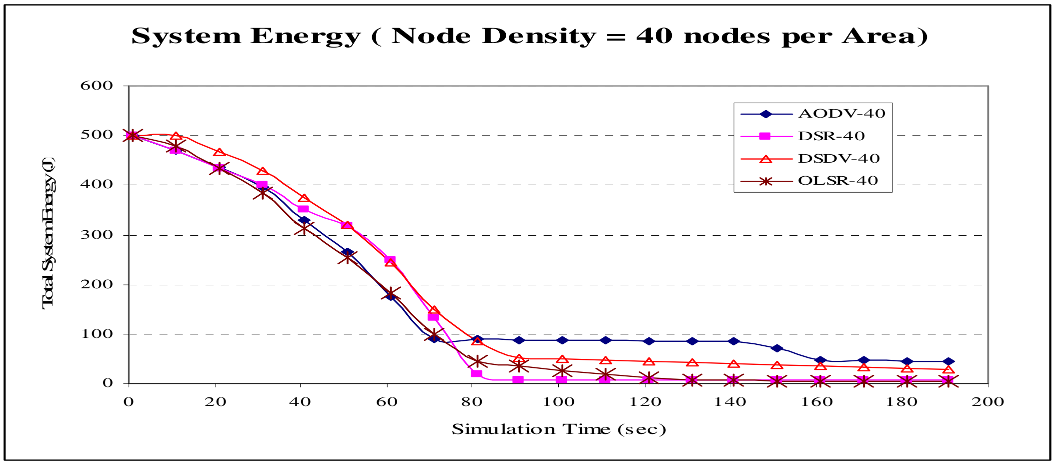


Figure 30. System Energy for Node Density = 40 Nodes per Area

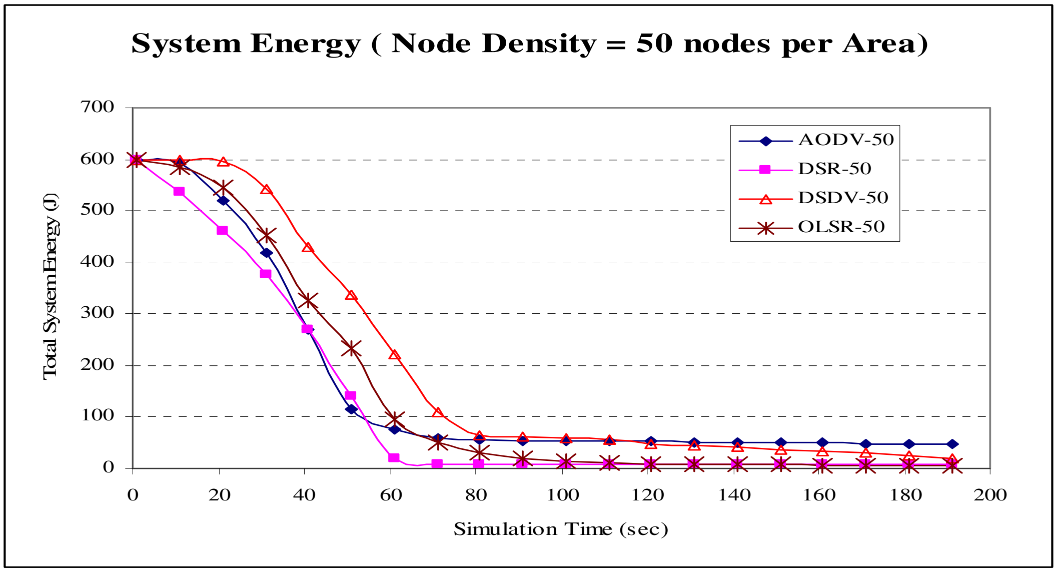


Figure 31. System Energy for Node Density = 50 Nodes per Area

3. Network Loading - Results

In this simulation, the connection load of each pair of communicating nodes was varied. The offered load is increased from 10 pkts/sec to 50 pkts/sec. Lower data rates can be seen as applications with lower bandwidth requirements, such as packetized voice and higher data rates, signify connections for more bandwidth-hungry applications like video streaming and large image transfer. The total number of nodes in this system remains at 20. Again, the rest of the simulation parameters are relatively unchanged and the speed of the nodes remained at 10m/s. The entire simulation run lasts for 200s. Table 9 summarizes the simulation parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	670m x 670m
Speed	10 m/s
Mobility Model	Random Waypoint
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10-20-30-40-50 pkts/sec
Pause Time	2s
Node Energy	10 Joules per node

Simulation Parameters	
Receive Power	300 mW
Transmit Power	800 mW
# of connections	10

Table 9. Simulation Parameters for Random Waypoint – Network Loading Variations

As shown in Figure 31, at a lower network load connection, AODV and DSR outperform OLSR. However, the case is reversed when the network loading increases to rates higher than 20 pkts/sec. OLSR has a better overall packet delivery ratio. Overall, the packet delivery ratio for the system drops approximately 30% when the traffic load connection increases five fold.

Figure 32 shows the routing overheads for all the routing protocols. OLSR and DSDV have relatively constant amount of overheads, independent of the network connection load while AODV and DSR increase by almost two fold as the network connection increases five fold.

For network latency, the results are plotted on graphs in Figure 33. DSDV has the best performance, i.e., the lowest average network latency at all connection rates while that of AODV and OLSR are very much close to one another. DSR's network latency increases significantly, as higher connection loading is approached.

For Figures 34 – 38, which indicate the energy performance of the four routing protocols as network loading increases, observe that DSDV has an initial better energy performance over OLSR when the loading is low. With the connection load getting heavier in each simulation run, OLSR emerges as a better candidate in conserving the system energy.

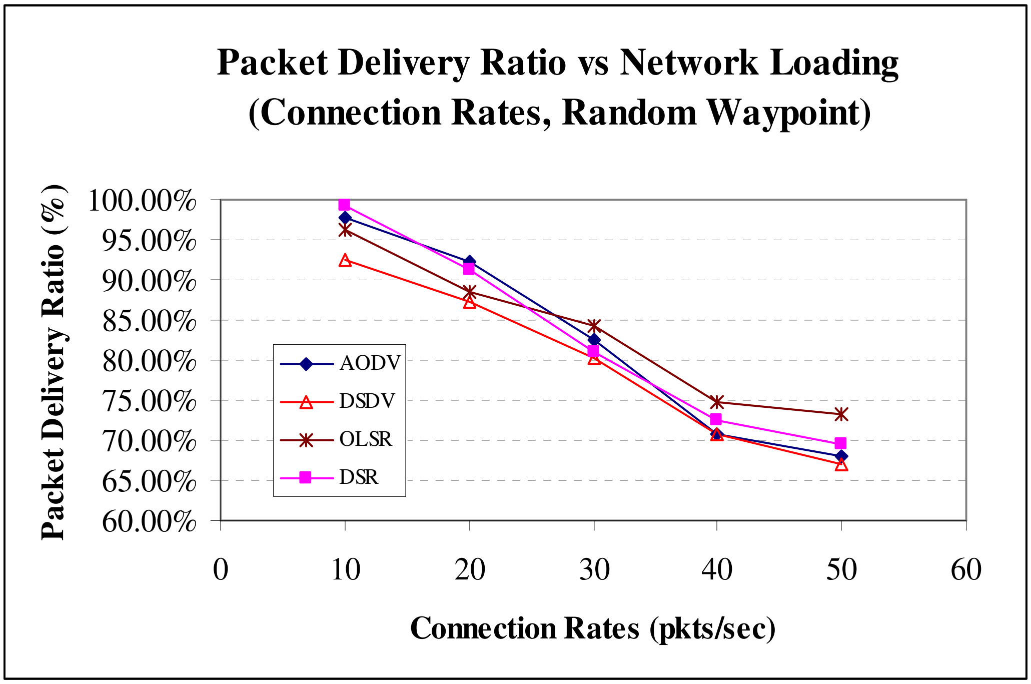


Figure 32. Packet Delivery Ratio with varied Network Loading (10-50 pkts /sec)

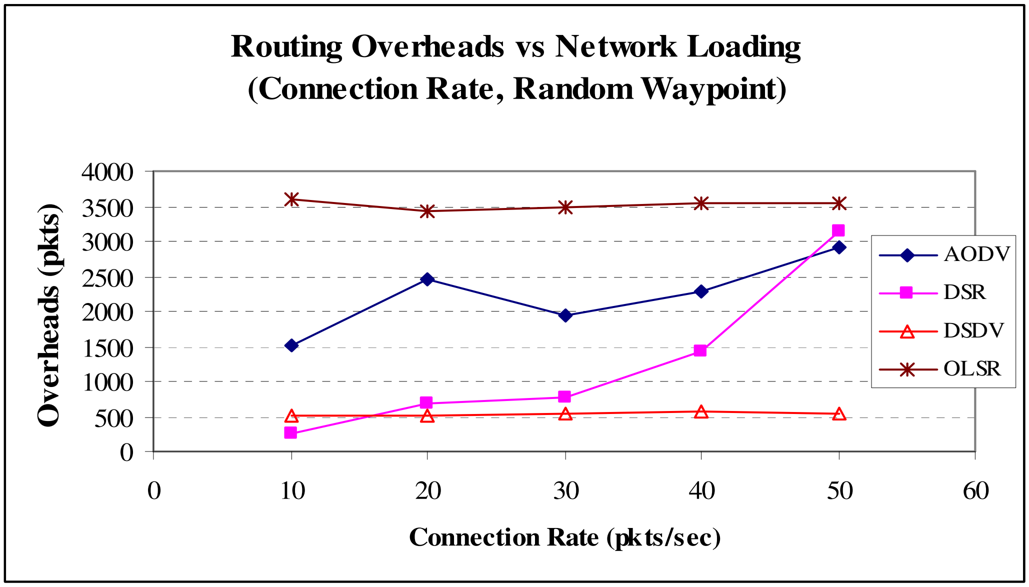


Figure 33. Routing Overheads with varied Network Loading (10-50 pkts /sec)

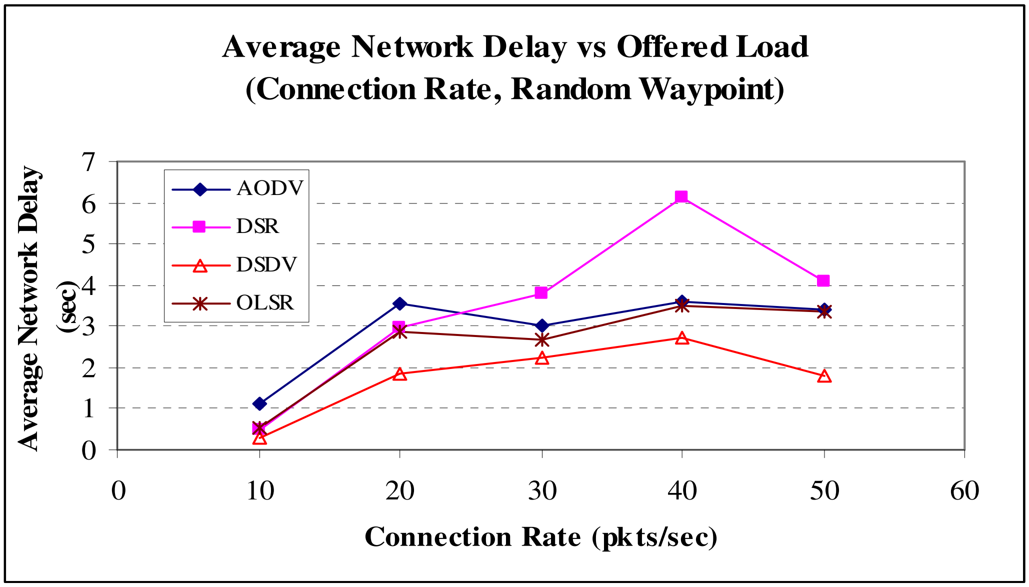


Figure 34. Average Network Delay with varied Network Loading (10-50 pkts /sec)

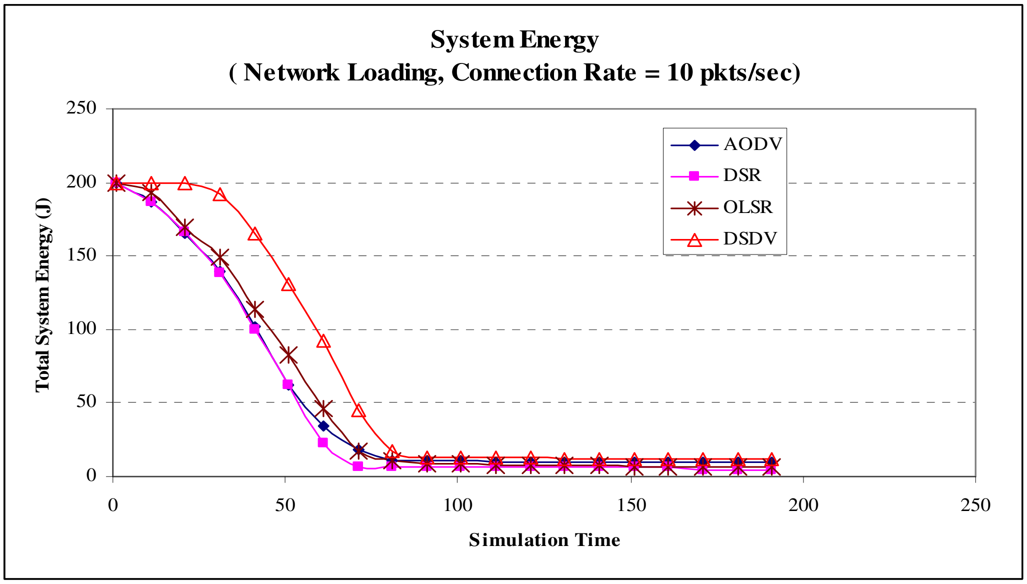


Figure 35. System Energy at Network Loading = 10 pkts/sec

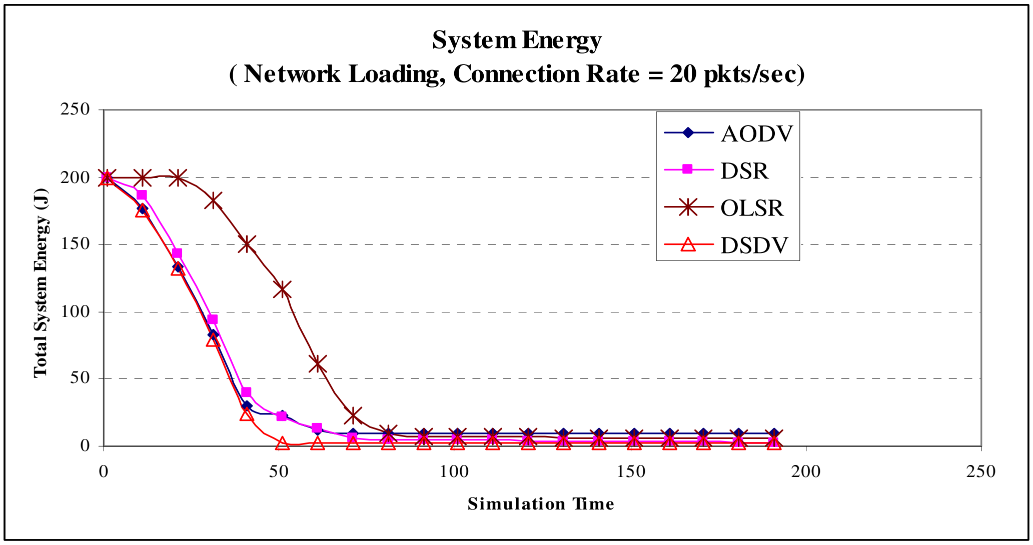


Figure 36. System Energy at Network Loading = 20 pkts/sec

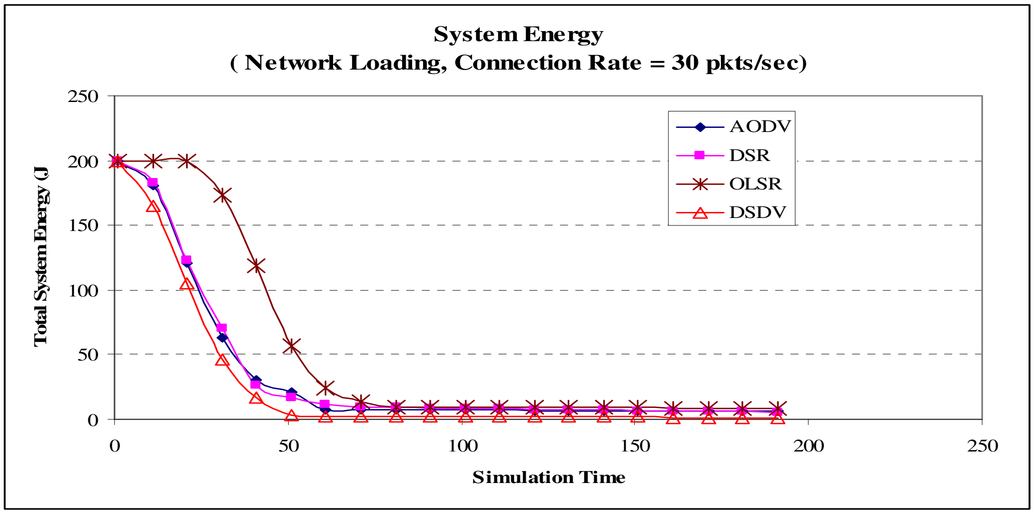


Figure 37. System Energy at Network Loading = 30 pkts/sec

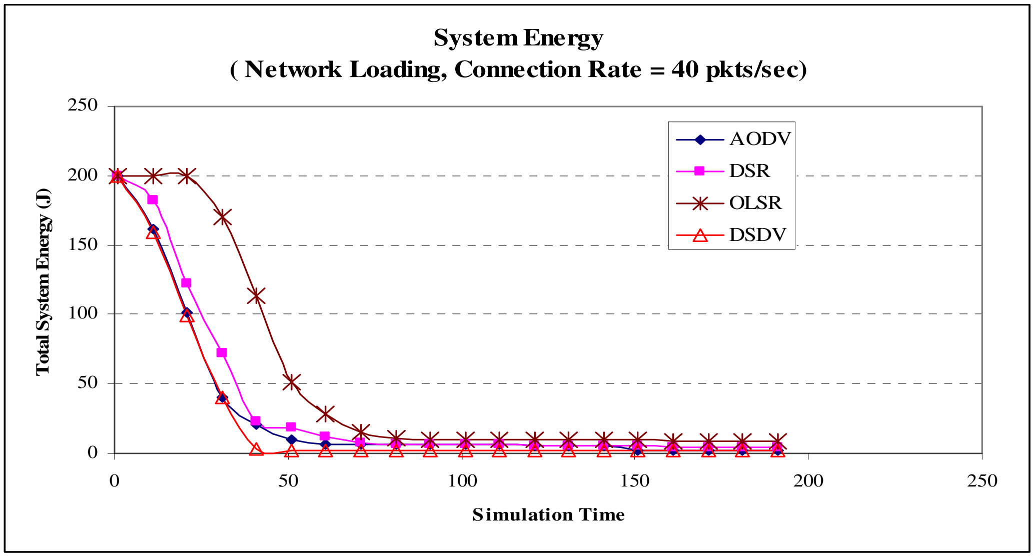


Figure 38. System Energy at Network Loading = 40 pkts/sec

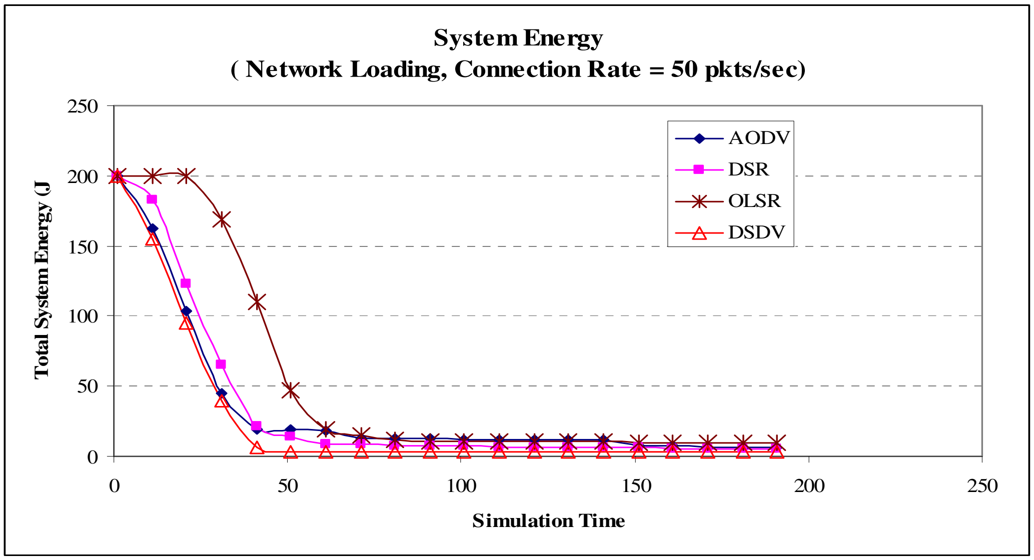


Figure 39. System Energy at Network Loading = 50 pkts/sec

In this second simulation of network load variation, the number of connections the simulated system will take was varied instead. The number of connections increases from five to 18 connections in each different sets of simulation. CBR was used as the traffic type and the connection rate (offered load) of each of these connections is kept at 10 pkts/sec. The total number of nodes in this system is 20. Again, the rest of the simulation parameters are relatively unchanged. Table 9 summarizes the simulation parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	670m x 670m
Speed	10 m/s
Mobility Model	Random Waypoint
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10 pkts/sec
Pause Time	2s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5-10-15-18

Table 10. Simulation Parameters for Random Waypoint – Network Loading Variations

Figure 31 shows the packet delivery ratio graph. Consistent with the two previous experimentations, there are on-demand protocols, AODV and DSR outperforming OLSR and DSDV at all connection rates. However, their difference decreases as the connection rate (network load) increases.

The routing overheads generated by the routing protocols is plotted in Figure 40. Note that at the initial phase when the connections quantity is between eight to 10, OLSR has significantly more overheads than AODV and DSR. As the connections number further increases to 15, AODV exceeds OLSR in routing overheads. Note that the difference between that of DSR and OLSR also decreases as the connection number increases. DSDV remains lowest in the generation of routing overheads, but at the expense of poorer packet delivery ratio, since routes may not be as fresh as necessary for the high delivery rate.

Figure 41 shows the network latency. Table-driven routing protocols, OLSR and DSDV, have the lowest network latency compared to the on-demand protocols. However, at the low connection number, all four have almost similar network latency values. Their difference is almost three times as large a gap when the number of connections is set to maximum. System energy graphs can be found in Figures 41 – 45. Note that AODV and DSR use up the system energy at a significant faster pace at the initial first 100 sec of the simulation. Towards the end of the simulation, all routing protocols will converge as nodes become exhausted after extensive routing and forwarding of data. OLSR and DSDV consume at a slower rate than AODV and DSR. DSR has the worst energy performance, i.e., consumes the total system energy the fastest.

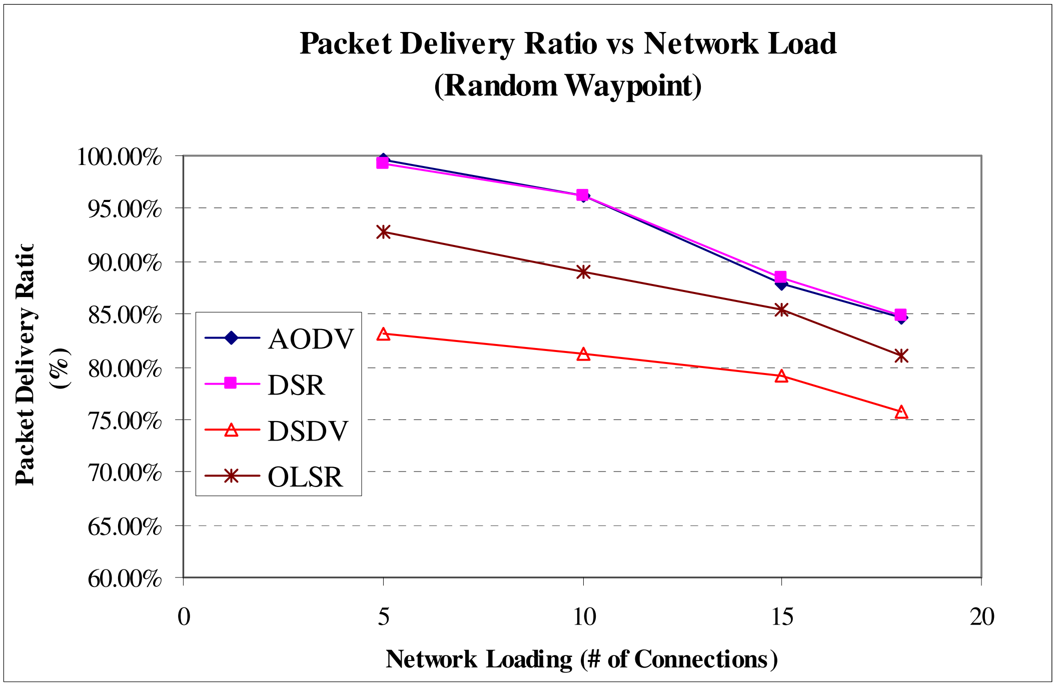


Figure 40. Packet Delivery Ratio with varied Network Loading (5-18 connections)

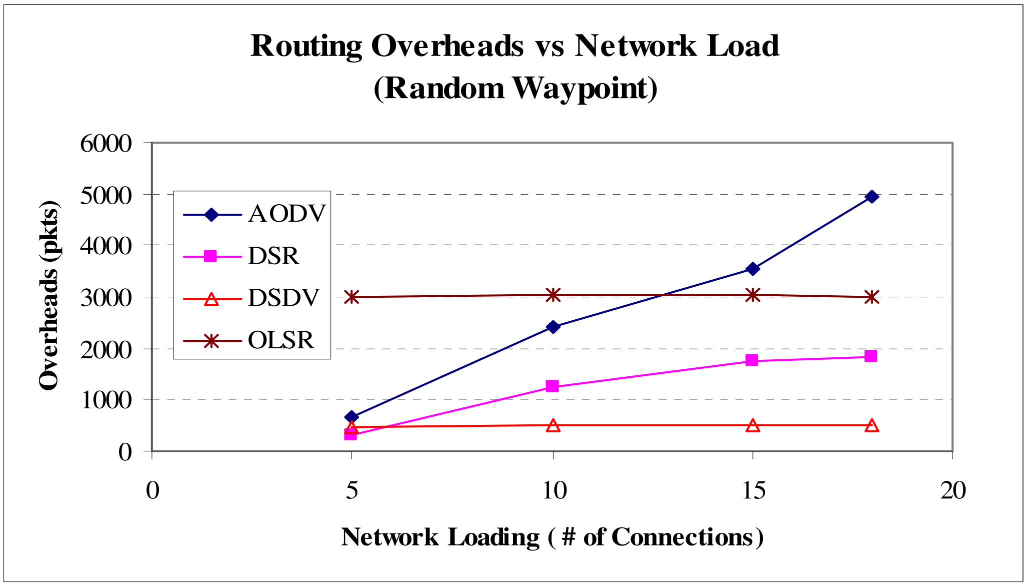


Figure 41. Routing Overheads with varied Network Loading (5-18 connections)

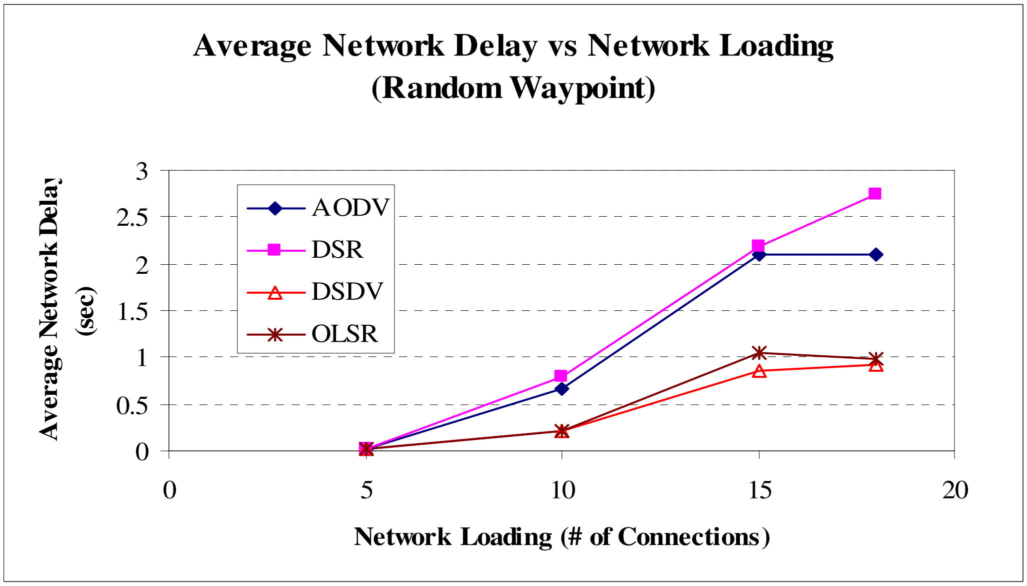


Figure 42. Average Network Delay with varied Network Loading (5-18 connections)

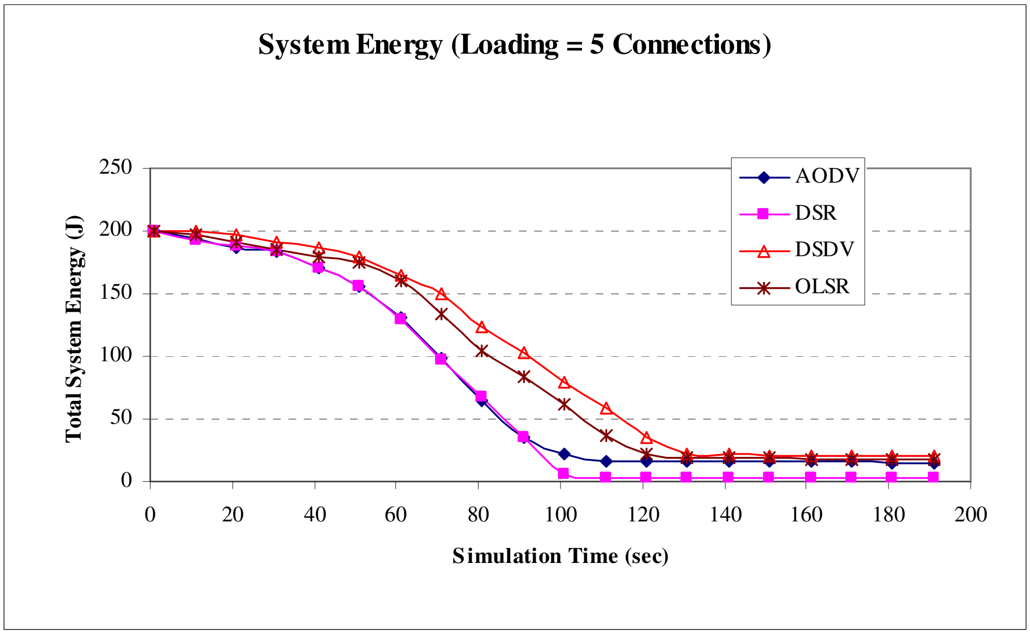


Figure 43. System Energy at Network Loading = 5 connections

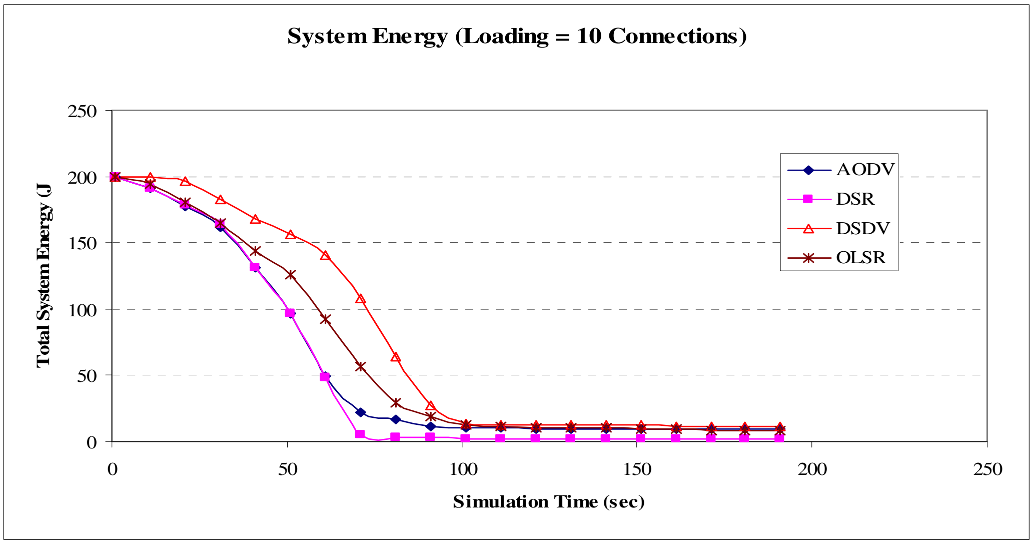


Figure 44. System Energy at Network Loading = 10 connections

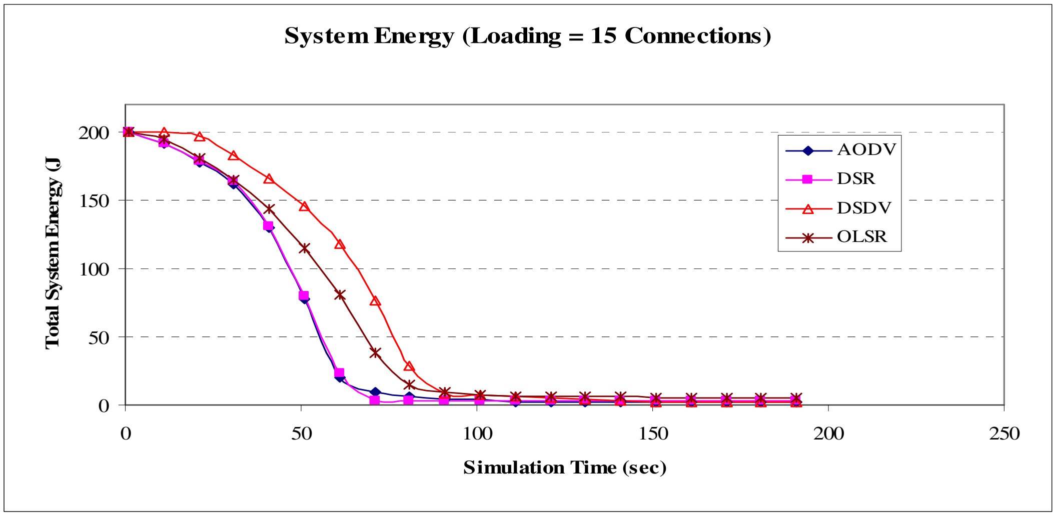


Figure 45. System Energy at Network Loading = 15 connections

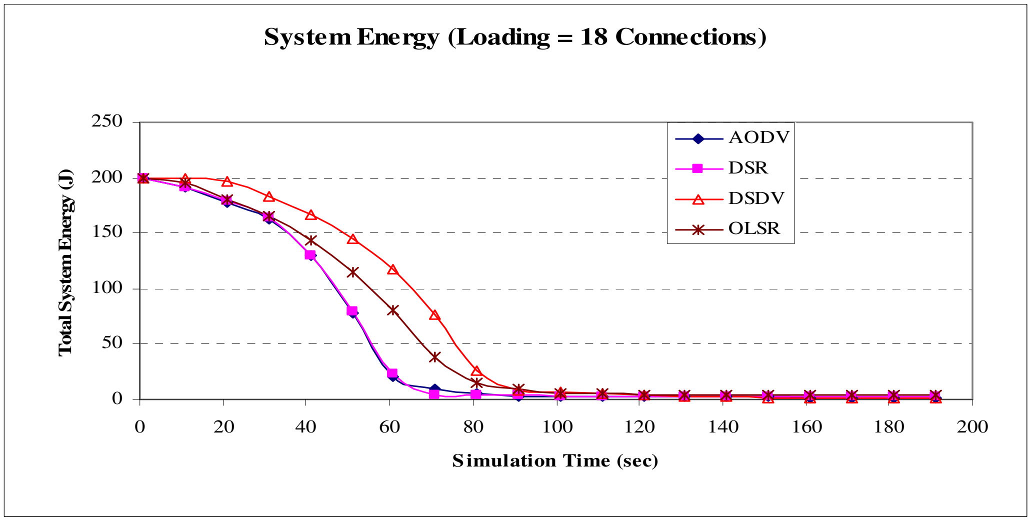


Figure 46. System Energy at Network Loading = 18 connections

B. MANHATTAN GRID MOBILITY MODEL

1. Mobility - Results

In the Manhattan Grid model, as described in the previous chapter, paragraph E.2, a bigger map area size is used due to the need to divide the map areas into multiple grids (10 by 10) for a 1000m x 1000m area. All things being equal as in the Random Waypoint model, similar parameters were used for the simulation. The first set of experiments is repeating that of the impact of mobility (the *mean* of the speed varying from 5 to 25 m/s) on the metrics' performance in consideration of the Manhattan Grid model instead. Table 11 summarizes the parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	1000m x 1000m
Speed	5-10-15-20-25m/s
Mobility Model	Manhattan Grid
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10 pkts/sec
Pause Time	10s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 11. Simulation Parameters for Manhattan Grid – Mobility Variations

Figure 46 shows the overall packet delivery ratio performance of the four routing protocols. Unlike the Random Waypoint model, OLSR outperforms DSR, DSDV and AODV. In fact, AODV has the worst performance in this scenario. DSR is better than DSDV as the speed increases but it almost has the same PDR at the first two speeds (5 and 10 m/s). In the Manhattan Grid models, nodes do not move diagonally. Instead, they travel either forward or backwards, right or left, and all directions are orthogonal to each other. The chance of a traffic signal breaking up increases as two nodes diverge. Hence, there are more route error messages due to disconnections. Note from Figure 47 that the routing overheads generated by AODV is greater than DSR. This can be attributed to more routing updates needed in AODV. DSR uses source routing and also caches some routing entries. Again, as in the case of the Random Waypoint model, OLSR, like DSDV, generates a significant amount of almost constant routing overheads despite the increase in mobility.

The average network delay for all routing protocols decreases as the mobility of the nodes increases from 5 to 25 m/s, as shown in Figure 48. This can be explained by the model as all the nodes are converging towards a single reference target point, and as such, the distance between these nodes decreases as speed increases since they would be closer to target and hence to each other's transmission range. The expectation is for better performance in the average network delay. Also, table-driven protocols have proven to be consistent in outperforming the on-demand protocols.

The energy consumption results are shown in Figures 49 - 53. With the exception of DSDV, all the remaining protocols roughly consume the system energy at the same rate since their curves are very close to one another. DSDV, in this respect, can be considered a more energy-efficient protocol. However, the trade off is a 2-3% drop in packet delivery ratio. Depending on the needs, it is possible to deploy either OLSR and DSDV in a situation that resembles more the Manhattan Grid model such as fighting in an urban setup.

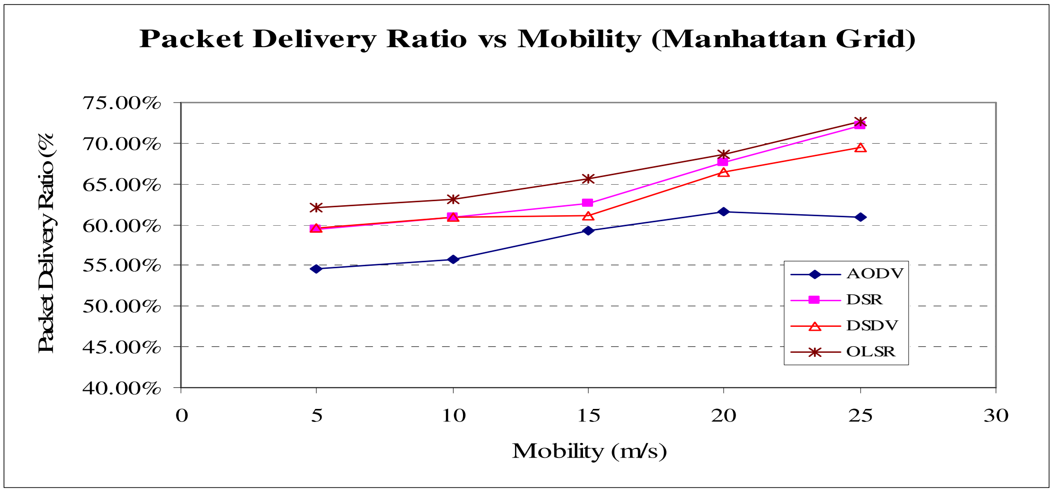


Figure 47. Packet Delivery Ratio with varied Speed (5 -25 m/s) – Manhattan Grid

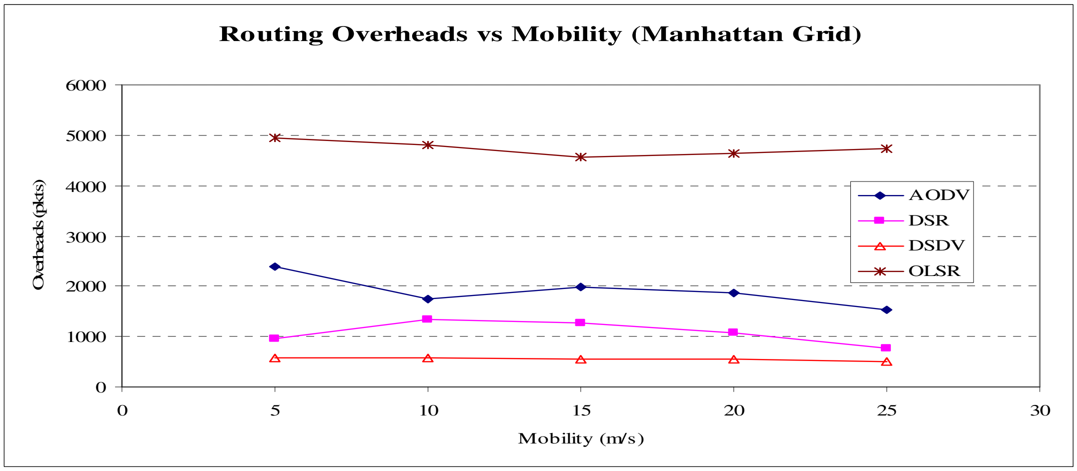


Figure 48. Routing Overhead with varied Speed (5 -25 m/s) – Manhattan Grid

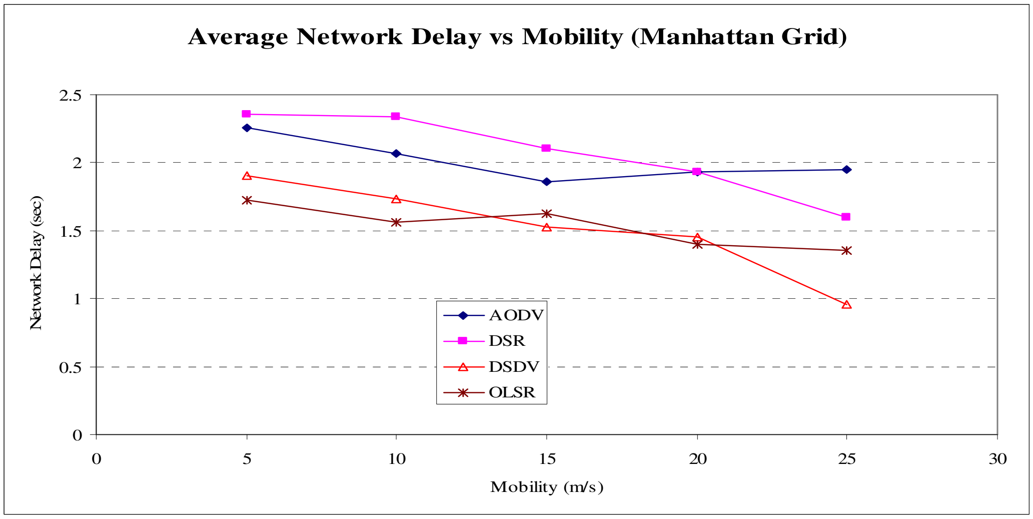


Figure 49. Average Network Delay with varied Speed (5 - 25 m/s) – Manhattan Grid

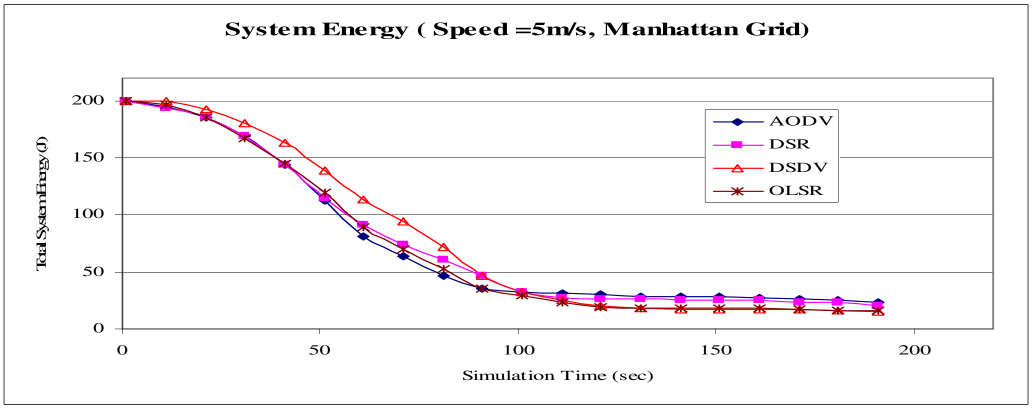


Figure 50. System Energy at Speed = 5 m/sec

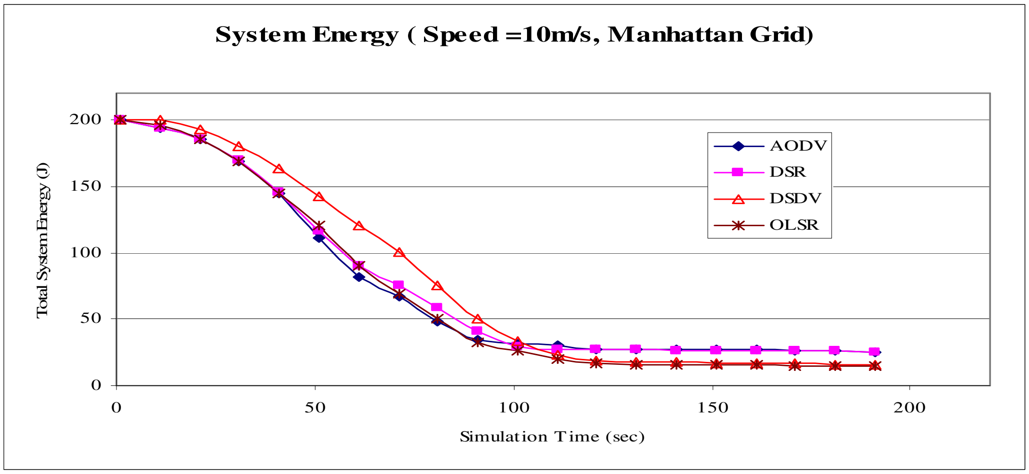


Figure 51. System Energy at Speed = 10 m/sec

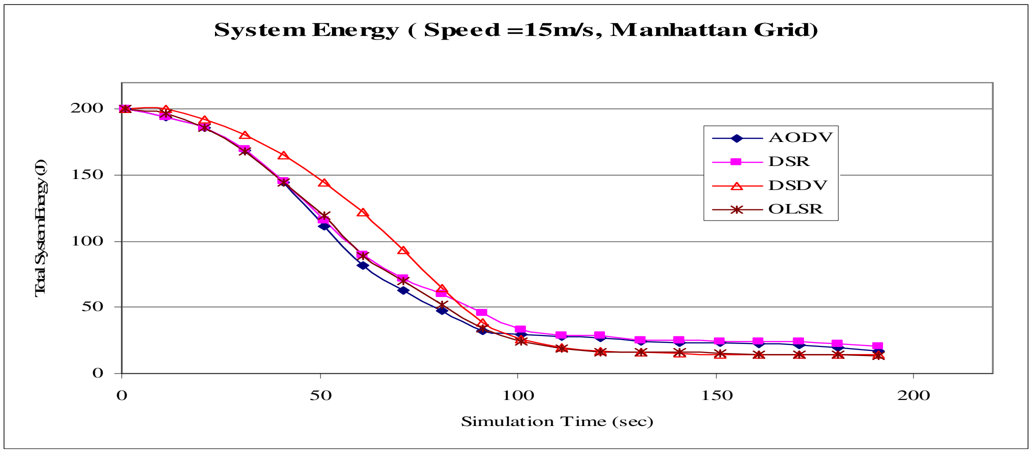


Figure 52. System Energy at Speed = 15 m/sec

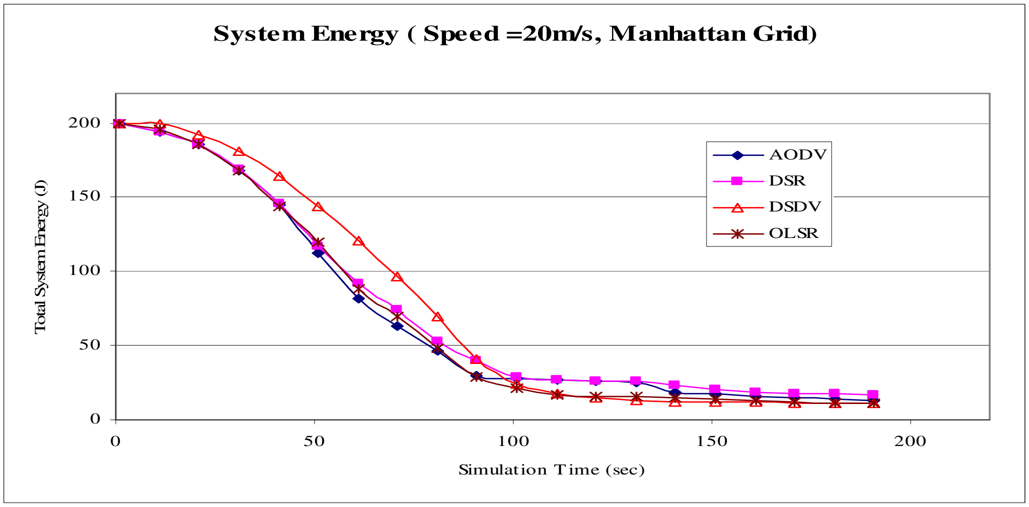


Figure 53. System Energy at Speed = 20 m/sec

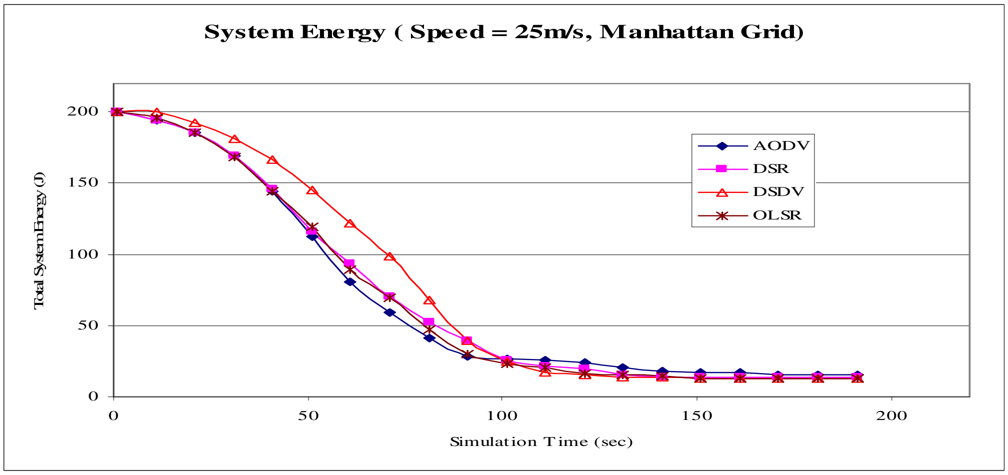


Figure 54. System Energy at Speed = 25 m/sec

2. Node Density - Results

In the second experiment on Manhattan Grid model, we investigate the performance aspect when the node density is varied within a fixed map area. The node density increases from 10 to 50 nodes within the same map area of 1000m x 1000m. See Table 12 for summary of the parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	10-20-30-40-50
Map Size	1000m x 1000m
Speed	10 m/s
Mobility Model	Manhattan Grid
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	20 pkts/sec
Pause Time	10s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 12. Simulation Parameters for Manhattan Grid – Node Density Variations

The results shown in Figure 54 illustrate the packet delivery ratio against the node density and in Figure 55 demonstrate the routing overhead incurred in this variation. Also, Figure 56 is the overall network delay of the system as the node density varies.

As seen in Figure 54, the packet delivery ratio began with a considerable variation when the node density is 20. AODV is almost twice that of DSDV. The difference slowly decreases as it become only about 8% upon approaching the 50 node density mark. Also, AODV, OLSR and DSR have very competitive performances and are relatively close in performance throughout the variation of node density. DSR actually starts to have degrade in PDR performance upon approaching a higher node density.

The routing overheads remain relatively low for DSR and DSDV, whereas that of AODV and OLSR increases tremendously by more than four and five fold, respectively. This is adverse for the network, as the goal is to not have the network be overly congested with too many control packets.

The network delay of DSR when compared with the rest has a significant order of magnitude difference of almost five to six times more. This is highly undesirable for delay sensitive applications. Hence, for scenarios similar in structure to the Manhattan Grid, it would be advisable not to use DSR for routing delay sensitive applications. DSDV and OLSR have the best network delay performance.

In the case of energy consumption, DSDV stays relatively ahead of the rest when the node density is relatively lower. At higher density nodes, DSR and AODV perform better in terms of less energy being consumed at the system level, although this only start to occur half way through the simulation.

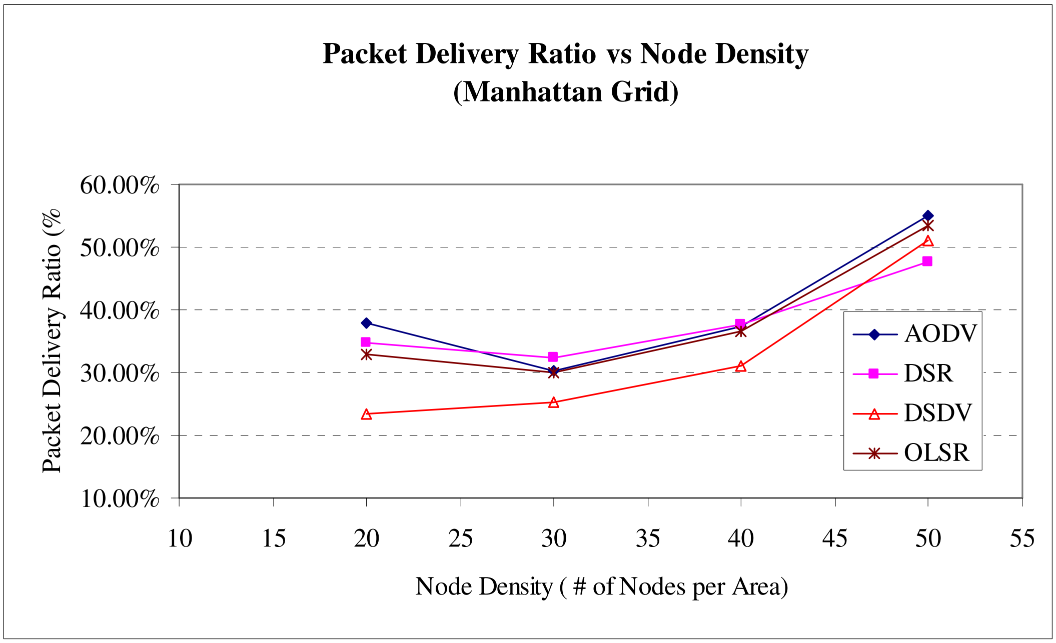


Figure 55. Packet Delivery Ratio with varied Node Density (20 – 50) – Manhattan Grid

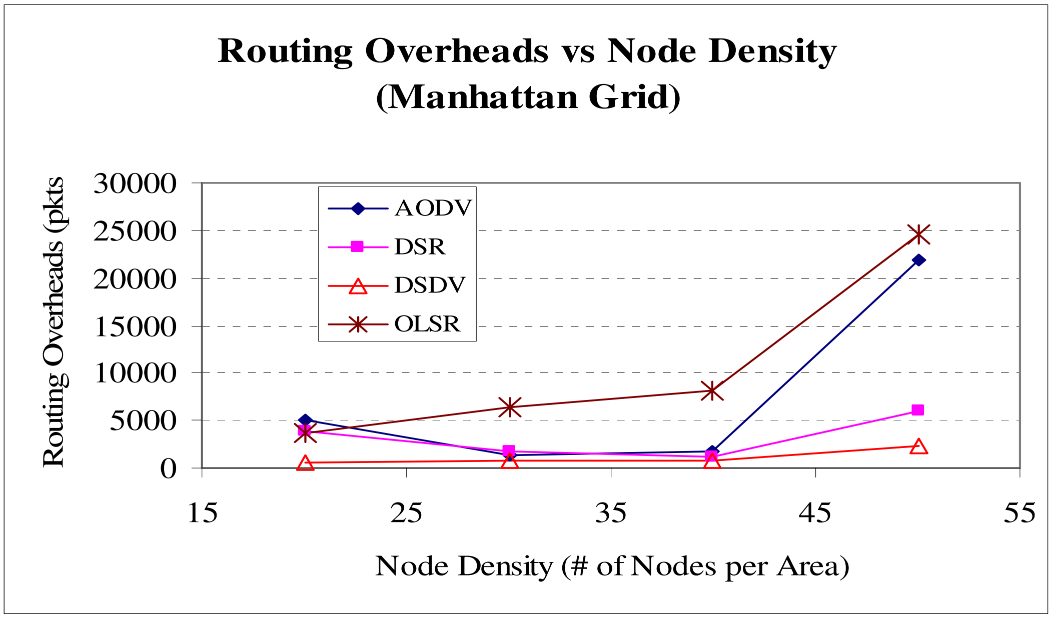


Figure 56. Routing Overhead with varied Node Density (20 – 50) – Manhattan Grid

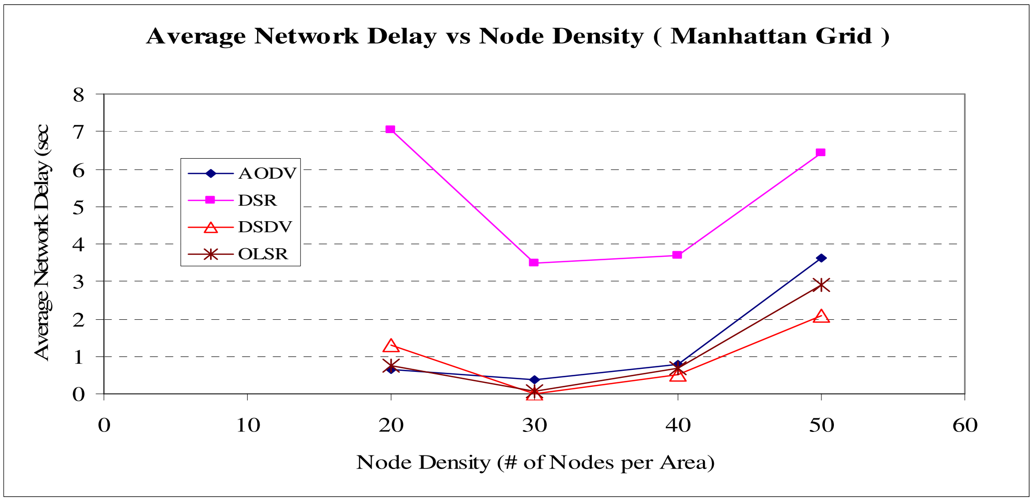


Figure 57. Average Network Delay with varied Node Density (20 – 50) – Manhattan Grid

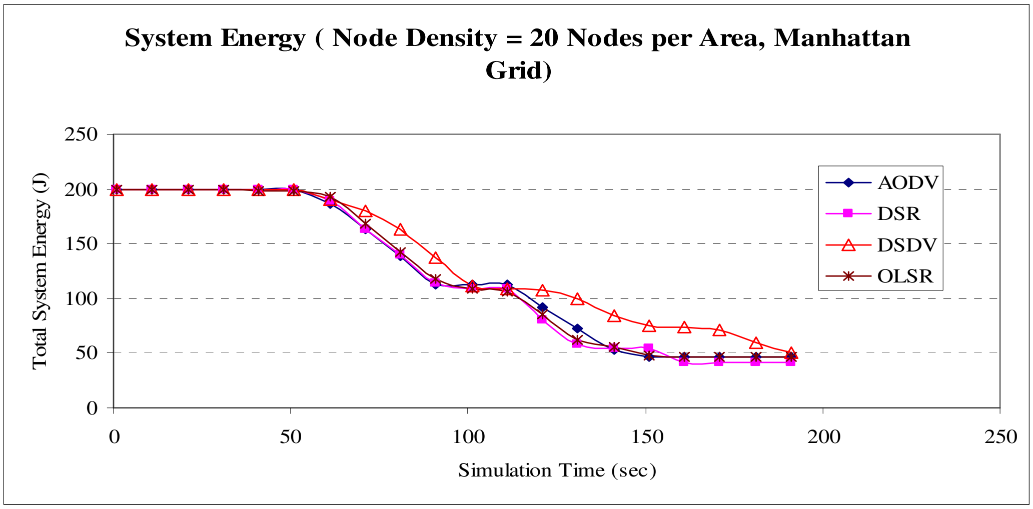


Figure 58. System Energy at Node Density of 20 Nodes

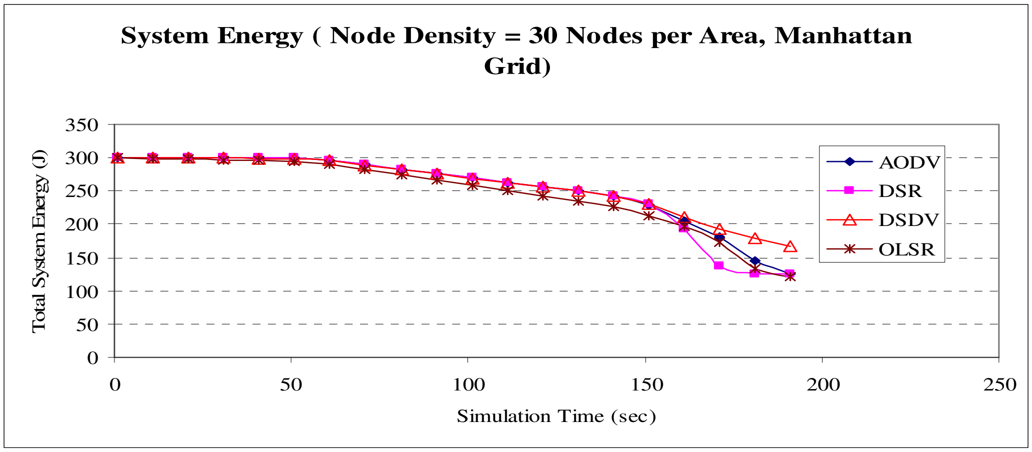


Figure 59. System Energy at Node Density of 30 Nodes

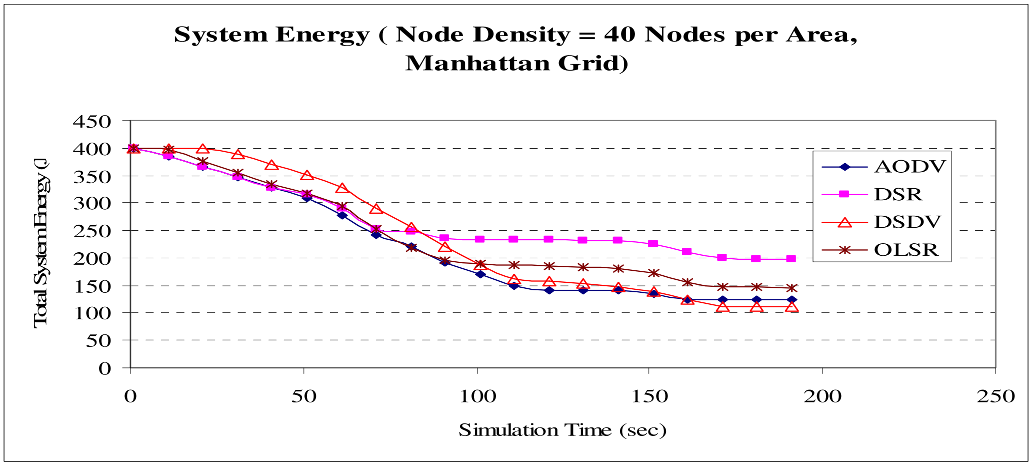


Figure 60. System Energy at Node Density of 40 Nodes

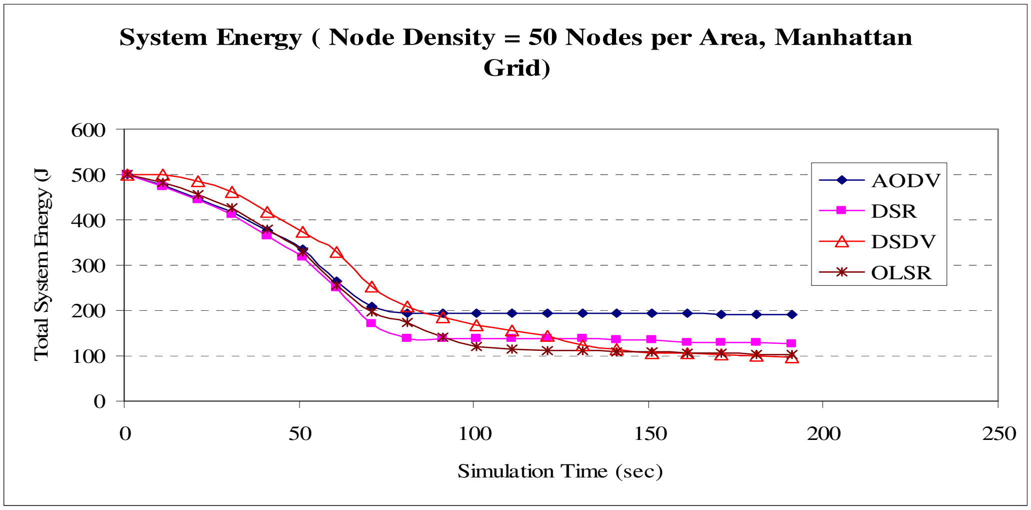


Figure 61. System Energy at Node Density of 50 Nodes

3. Network Loading - Results

In the final simulation on Manhattan Grid model, we investigate the routing performance aspect when the offered load increases. We do so by increasing the average connection load offered by each connection, starting at 10 pkts/sec to 50 pkts/sec. This study of increase in network loading is to understand the impact of bandwidth intensive applications such as video streaming. See Table 13 for summary of the parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	1000m x 1000m
Speed	10 m/s
Mobility Model	Manhattan Grid
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10-20-30-40-50 pkts/sec
Pause Time	10s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 13. Simulation Parameters for Manhattan Grid – Network Loading Variations

Figure 61 shows the packet delivery ratio for Manhattan Grid modeling with variations in network load. Figures 62 and 63 indicate the routing overheads as well as the network delay of this simulation.

Figure 61 illustrates the packet delivery ratio of routing protocols and demonstrates that all protocols have very close performance outputs especially when the network loading is high. DSDV has poorer performance when the initial traffic intensity is lower, and therefore, it is possible to deduce that as far as packet delivery ratio is concerned, not much difference exists between the routing protocols used.

Figure 62 presents the routing overheads Again, OLSR having a significant factor of approximately five times more overheads over DSDV and three to four times more over DSR and AODV. Although, as the network loading increases, note that the on-demand protocols tend to increase their overheads, although not enough to exceed that of OLSR.

The network delay graph in Figure 63 indicates that as the network loading increases, a dramatic increase of the average network delay experienced by DSDV nodes is witnessed. DSDV is a table-driven protocol in which nodes exchanges updates information regarding their connectivity. With a high network loading at the same time, many such route updates are unable to get through, thereby, causing route information to be slow to establish. As such, packets obtain buffers as route tables are being established at the same time. In the OLSR case, only designated MPRs are sending routes, and hence, significantly reducing the route exchanges, and as such, the routing tables are built faster. This is the main advantage of OLSR over DSDV.

System energy variations, as shown in Figures 64 to 68 in different loading cases, suggest that DSDV has only a significant impact at the start of the simulation when DSDV nodes are conserving energy to transmit data. However, with overwhelming network loading towards the second half of the simulation, all protocols consume the energy of the nodes quickly as routes are contending with data at the same time for the same bandwidth, which leads to the “convergence” of the graphs.

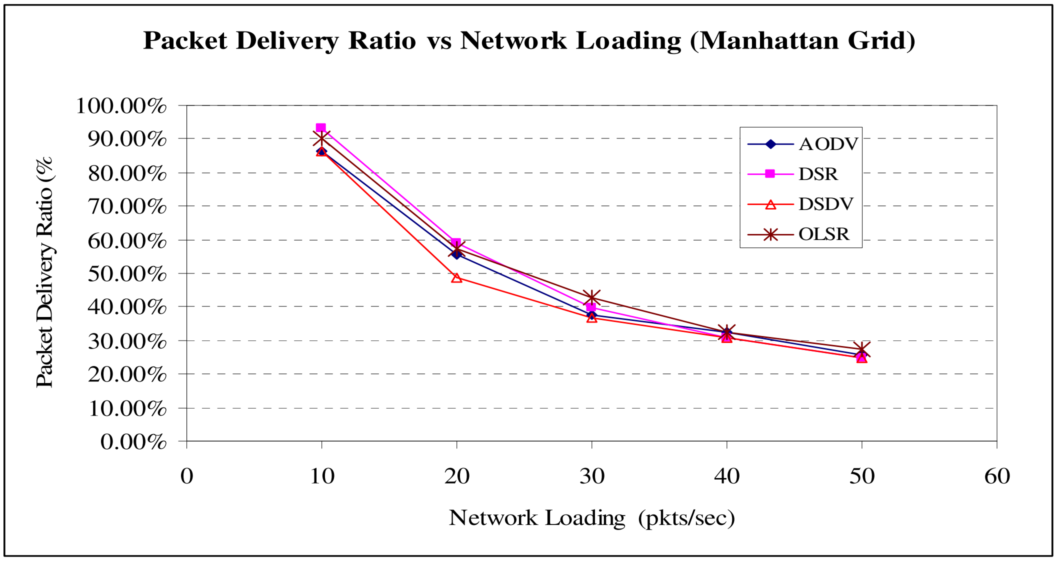


Figure 62. Packet Delivery Ratio with varied Network Loading (10- 50 pkts /sec) – Manhattan Grid

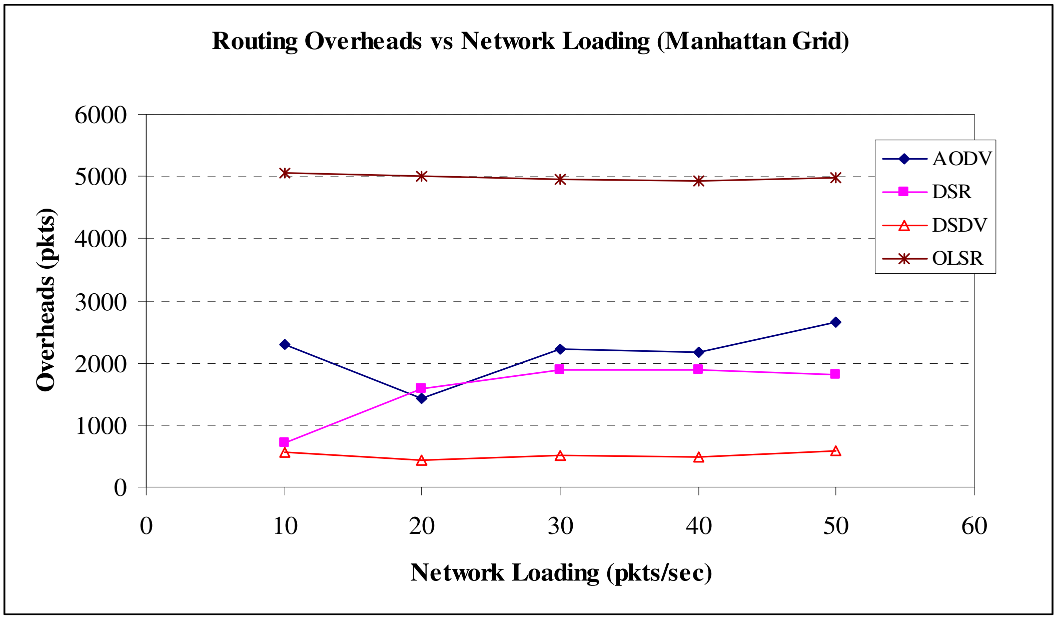


Figure 63. Routing Overheads with varied Network Loading (10- 50 pkts /sec) – Manhattan Grid

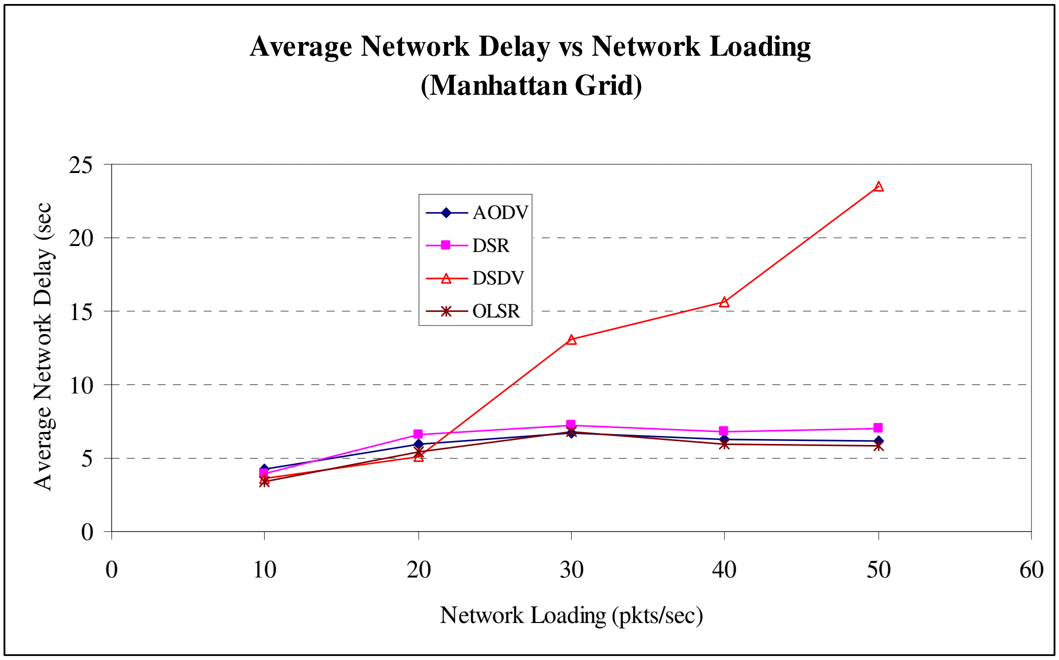


Figure 64. Average Network Delay with varied Network Loading (10 - 20 pkts/sec) – Manhattan Grid

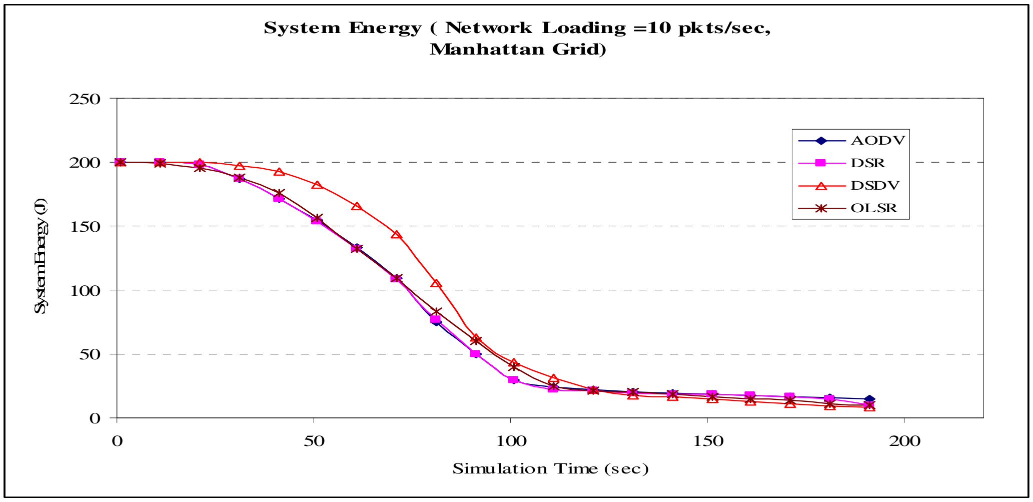


Figure 65. System Energy at Network Loading = 10 pkts/sec

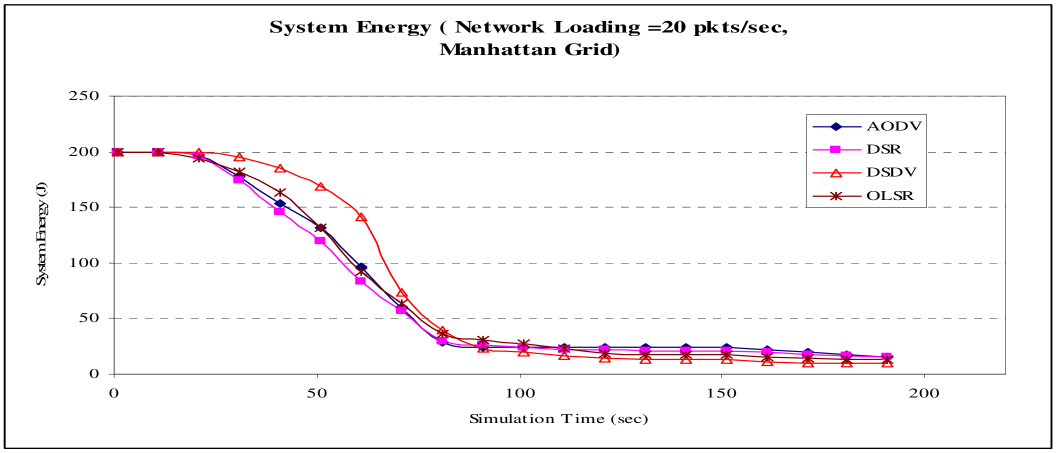


Figure 66. System Energy at Network Loading = 20 pkts/sec

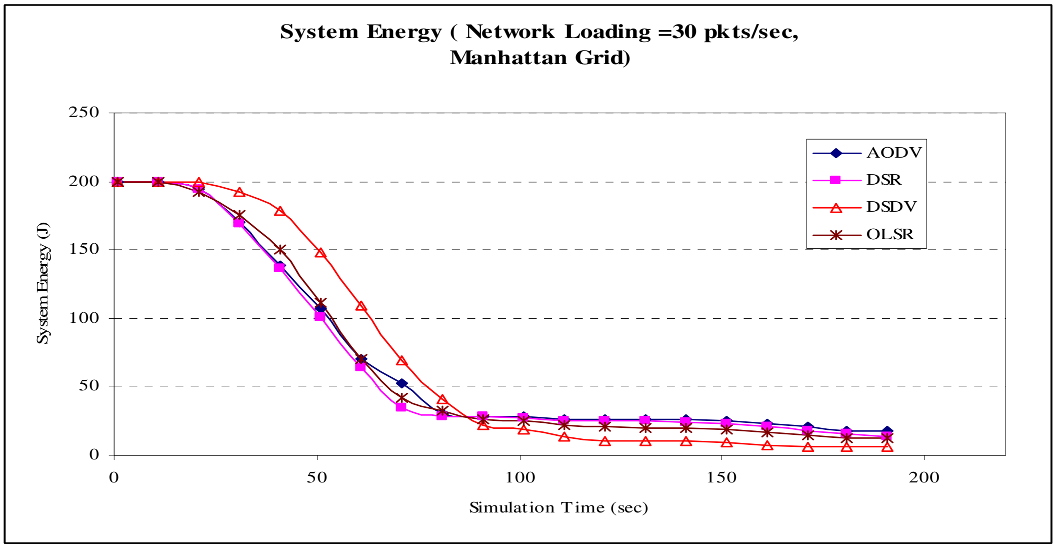


Figure 67. System Energy at Network Loading = 30 pkts/sec

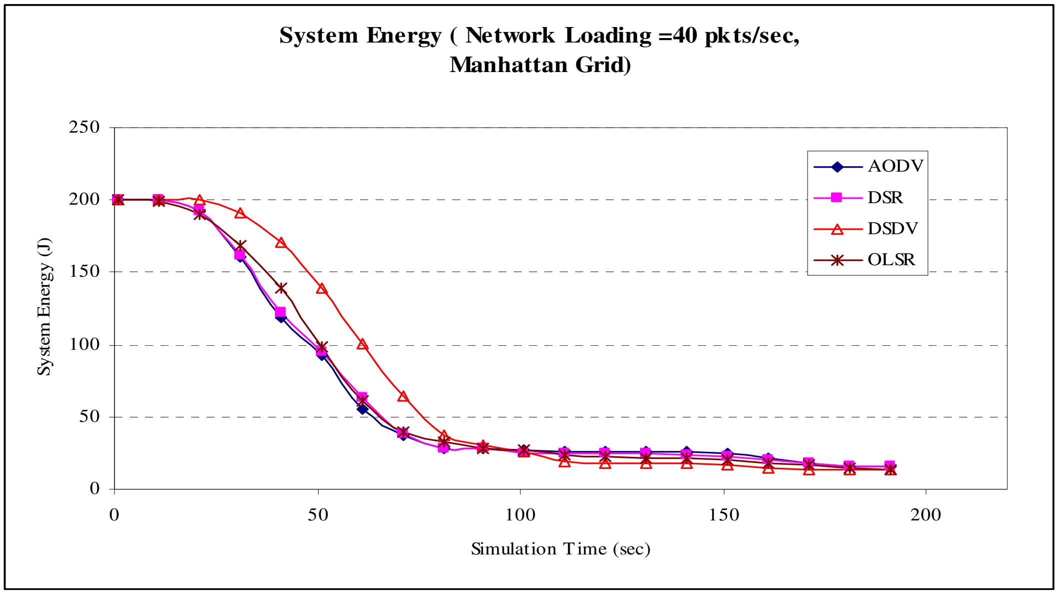


Figure 68. System Energy at Network Loading = 40 pkts/sec

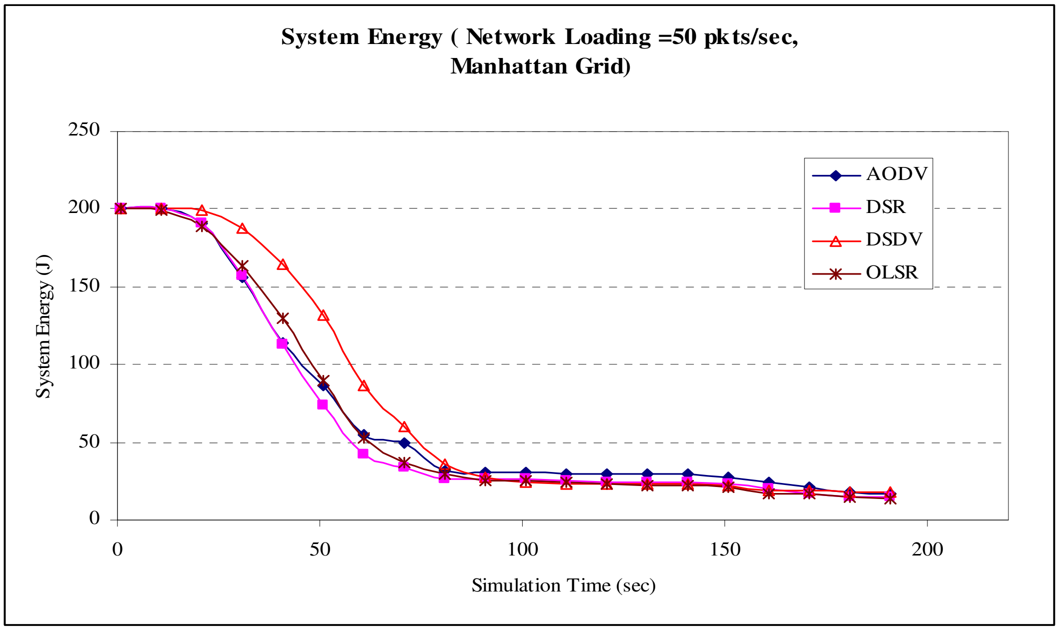


Figure 69. System Energy at Network Loading = 50 pkts/sec

C. REFERENCE POINT GROUP MOBILITY MODEL

1. Mobility - Results

In the Reference Point Group Mobility model (RPGM), as described in the previous chapter, paragraph E.3, nodes cluster together and move as a group. All things being equal as in the Random Waypoint model, similar parameters for simulation are used. The first set of experiments is repeating that of the impact of mobility (the *mean* of the speed varying from 10 to 25 m/s) on the metrics' performance using the RPGM model. See Table 14 for a summary of the parameters.

The packet delivery ratio graph is plotted in Figure 69, which suggests that the AODV, DSR and OLSR have almost similar performance outputs. DSDV performs consistently worse than the rest with a difference in gap of approximately 10% to 15%. This is a significant difference as applications may suffer serious service degradation due to lower bandwidth available.

Figure 70 plots routing overheads. This indicates that the routing overheads is the lowest for DSDV. For AODV, the routing overheads declines significantly as the speed increases. This is due to the fact that nodes are converging towards the centre and the paths are consistently established between the nodes as the distance separating them decreases over the higher speed. OLSR remains high on routing overheads despite higher mobility.

Figure 71 displays the average network delay graph. Interestingly, DSDV's performance in network delay is the worst amongst the four. OLSR has the best performance and is followed by AODV. The high delay in DSDV could be due to the large update interval, as compared to OLSR hello and topology control intervals. Due to longer update periods, DSDV nodes may not have the newest routes, and therefore, a route discovery process must be launched, and thereby, slow down the overall transfer time of data. However, in the high speed mobility situation, the huge gap in network delay diminishes and is, in general, half a second longer than other routing protocols.

The energy consumption graphs are featured in Figures 72 to 75. DSDV has again demonstrated lower energy consumption rate at the beginning of the simulation. It, however, degrades quickly upon proceeding beyond the mid-point of the simulation. The rest of the protocols have relatively the same energy consumption rates with AODV and OLSR performing better at higher mobility scenarios.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	1000m x 1000m
Speed	10-15-20-25m/s
Mobility Model	Reference Point Group Mobility (RPGM)
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	10 pkts/sec
Pause Time	20-30s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 14. Simulation Parameters for RPGM – Mobility Variations

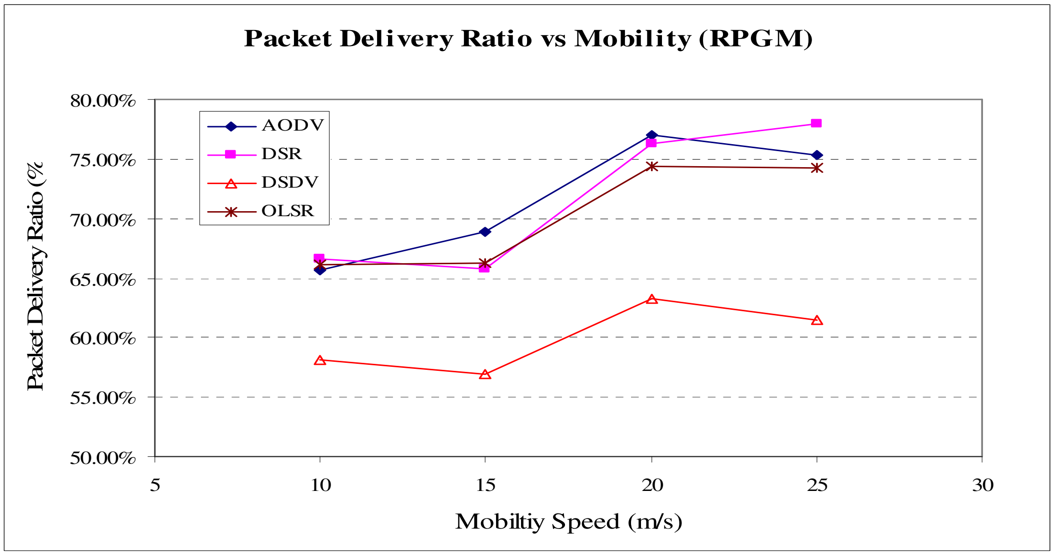


Figure 70. Packet Delivery Ratio with varied Speed (10 -25 m/s) – RPGM

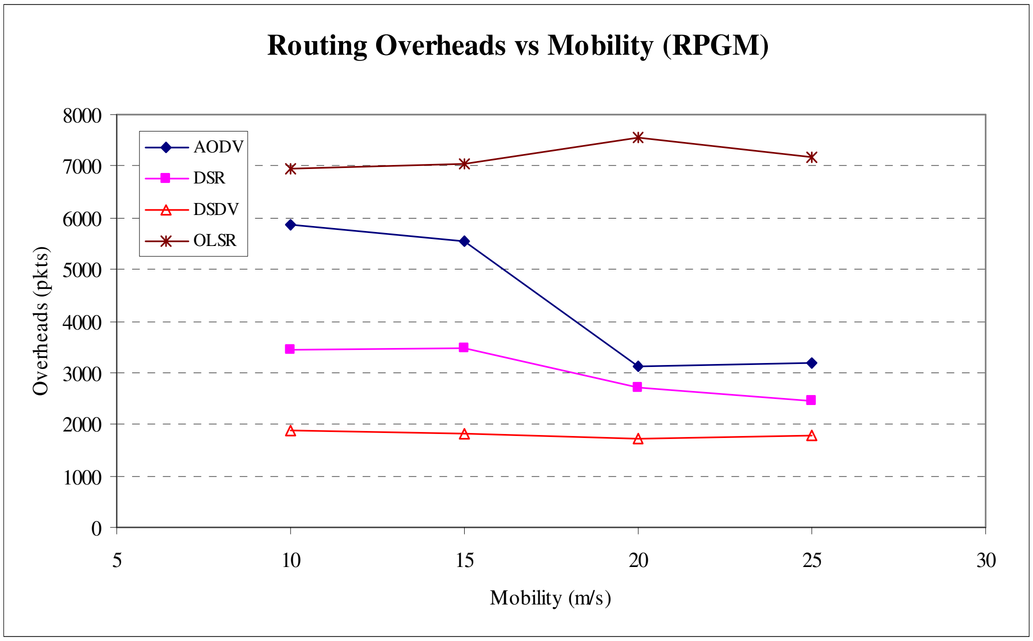


Figure 71. Routing Overhead with varied Speed (10 -25 m/s) – RPGM

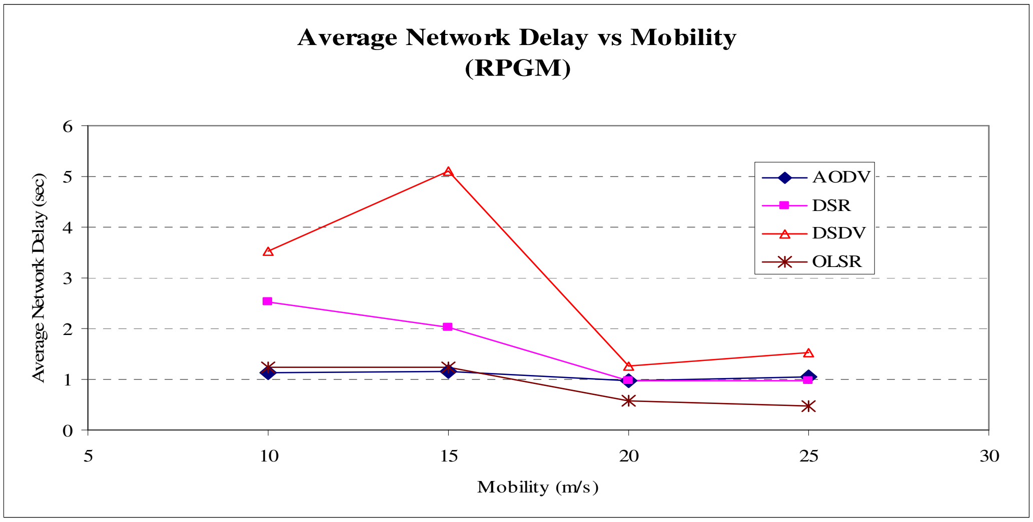


Figure 72. Average Network Delay with varied Speed (10 - 25 m/s) – RPGM

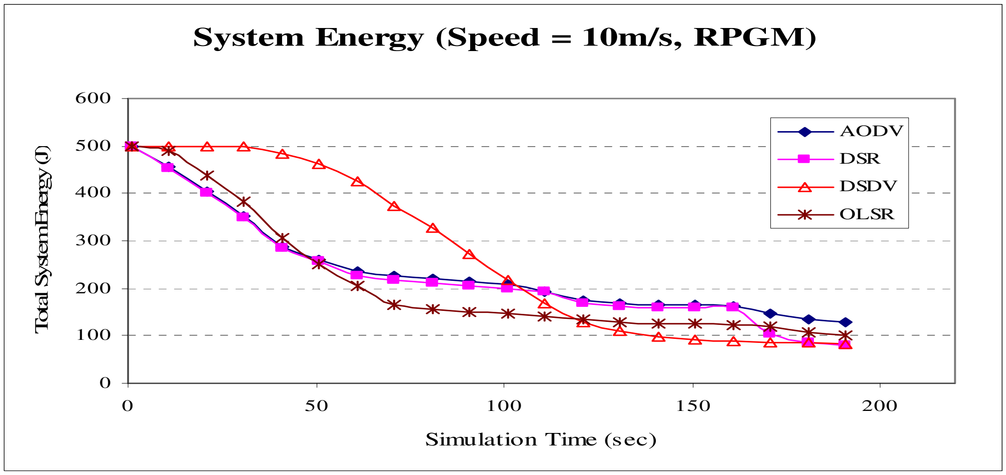


Figure 73. System Energy at Speed = 10m/s

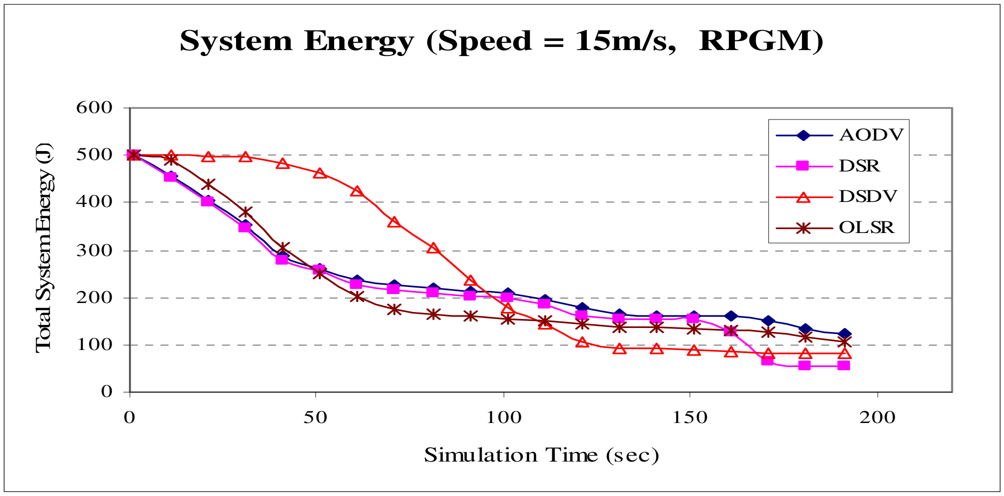


Figure 74. System Energy at Speed = 15m/s

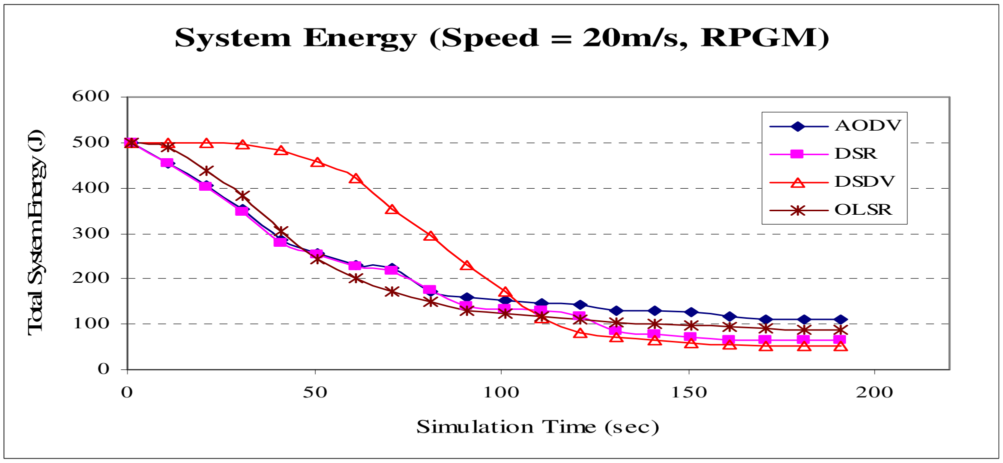


Figure 75. System Energy at Speed = 20m/s

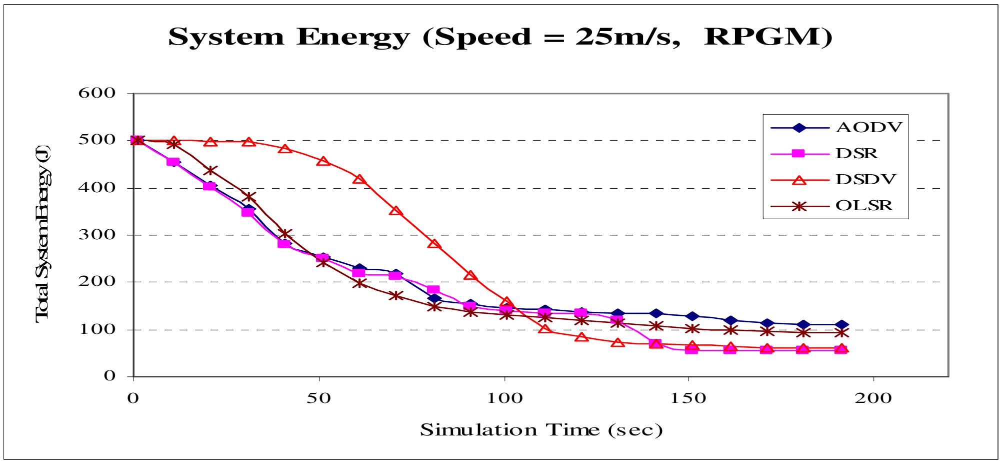


Figure 76. System Energy at Speed = 25m/s

2. Node Density - Results

In the second experiment of the RPGM model, the performance aspect is investigated when the node density is varied within a fixed map area. The node density is incrementally adjusted from 20 to 100 nodes within the same map area of 1000m x 1000m. The mobility is the same as described in the previous chapter, paragraph E.3. Nodes are kept at an average mean speed of 10 m/s sending at a rate of 20 pkts/sec between communication pairs. Table 12 summarizes the parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20-50-80-100
Map Size	1000m x 1000m
Speed	10 m/s
Mobility Model	Reference Point Group Mobility (RPGM)
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	20 pkts/sec
Pause Time	20-30s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 15. Simulation Parameters for RPGM – Node Density Variations

The results are shown in the series of graphs below. Figure 76 shows the packet delivery ratio of the network using the four different protocols in a RPGM model. Note that, overall, OLSR performance is the most outstanding amongst the three as the node density increases. DSDV also has a positive impact on the overall packet deliver ratio, catching up on-demand routing protocols as the node density increases.

Figure 77 highlights the overall routing overheads generated in the course of the simulation period by the four routing protocols. In the case of AODV and DSR, there is a significant increase, almost exponential in shape, as the node density increases from 20 to 100. This has a significant impact on the overall network delay, signifying that, as proven in Figure 78, the overall delay experienced by AODV and DSR nodes will be significant as many routes are generated. Traditionally, on-demand protocols are associated to scale better when the node number becomes larger. However, in this case, it says that on-demand protocols may *not* scale as well as table-driven protocols when the mobility pattern is that of RPGM, although there is a higher node density. As such, mobility patterns do have significant impacts on the overall performance of the network system using specific routing protocols.

Figure 77 demonstrates that OLSR has emerged as the routing protocol providing the lowest latency experienced by nodes within a fixed area. DSDV is second. However overall, OLSR has, in addition, better packet delivery ratio, although it still has a higher number of routing overheads. Note that deploying OLSR in such a scenario would be a better candidate than the rest.

Figures 78 – 82 indicate the overall system energy levels as the simulation passes through the different stages. DSDV remains a good candidate if energy consumption is the over-riding factor, and note that DSDV maintains, for quite a while at the first 100 sec of the simulation for a good level of total system energy, while OLSR zaps up the system energy at a significantly faster pace, especially when the node density is high. The shortcomings of this aspect of OLSR are caused by the large number of overheads generated between the nodes within a group. One method is to set the hello intervals at a higher intervals so as to lower the number of routes created.

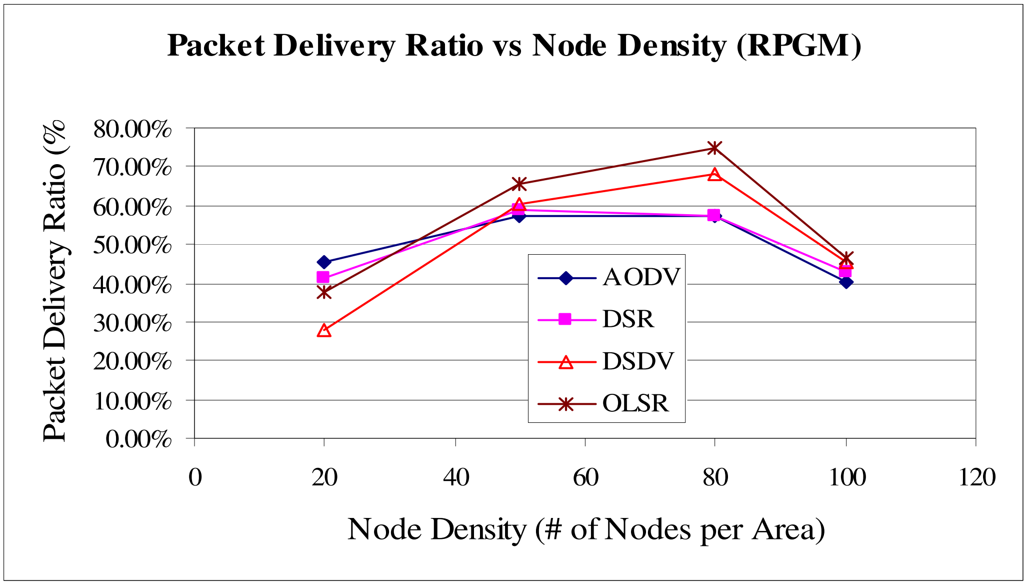


Figure 77. Packet Delivery Ratio with varied Node Density (20-100 nodes) – RPGM

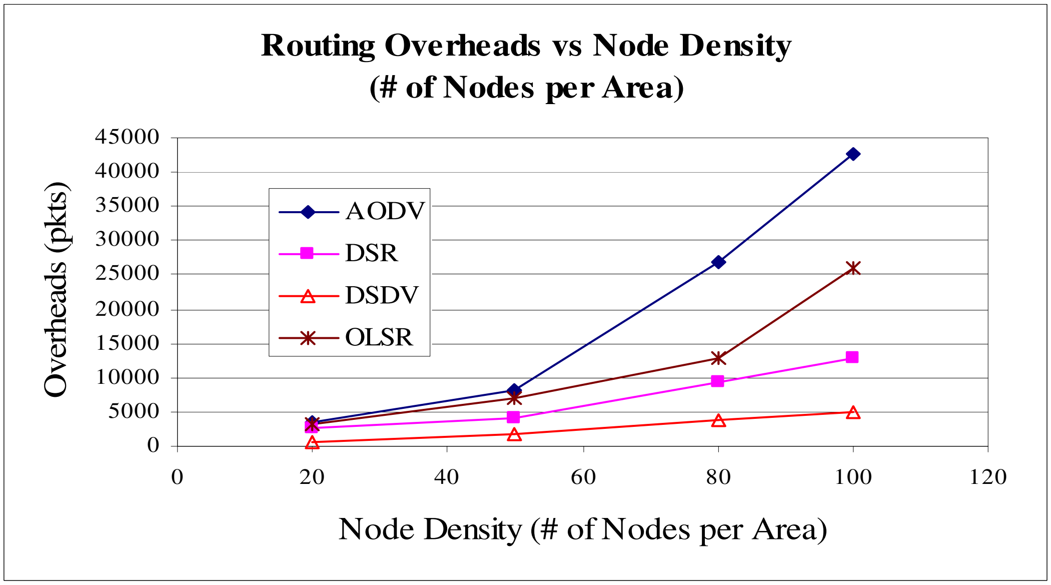


Figure 78. Routing Overhead with varied Node Density (20-100 nodes) – RPGM

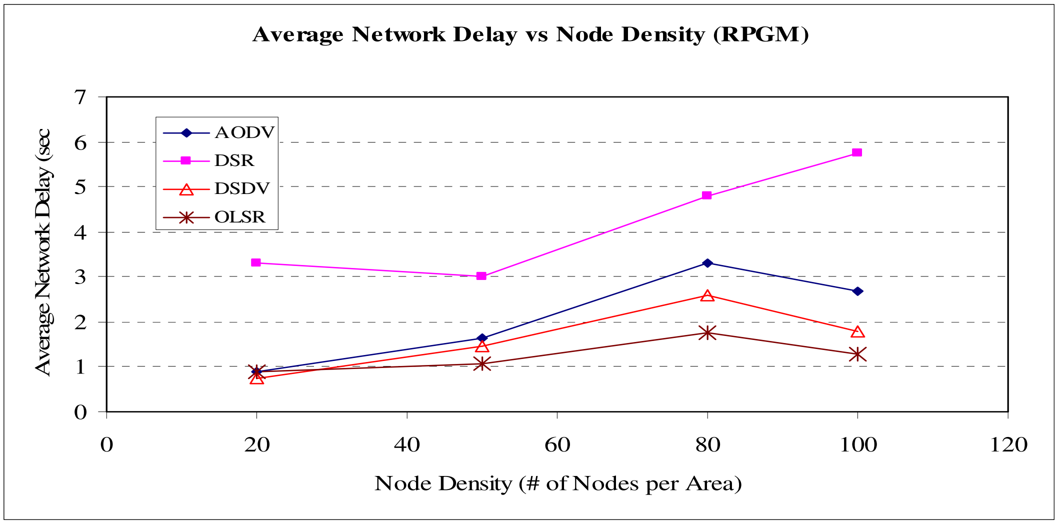


Figure 79. Average Network Delay with varied Node Density (20-100 nodes) – RPGM

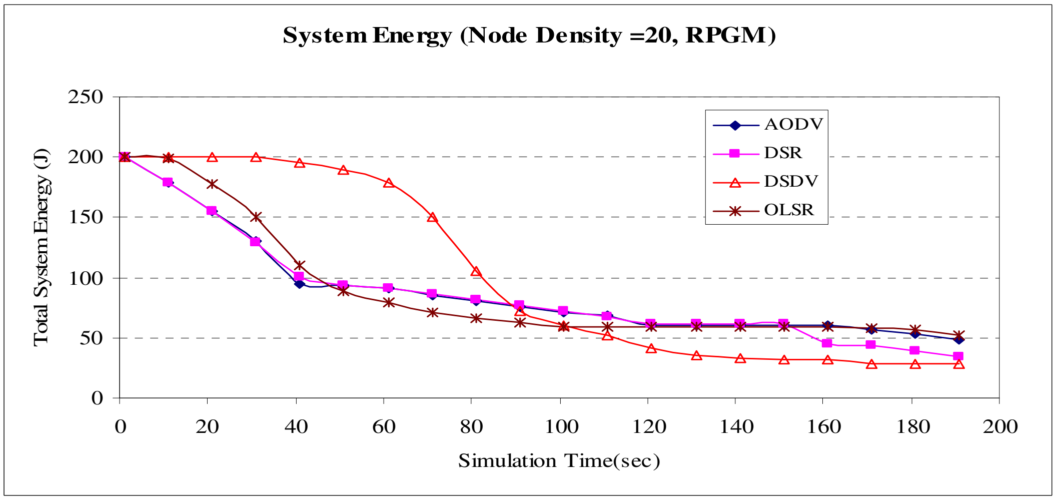


Figure 80. System Energy at Node Density = 20 Nodes

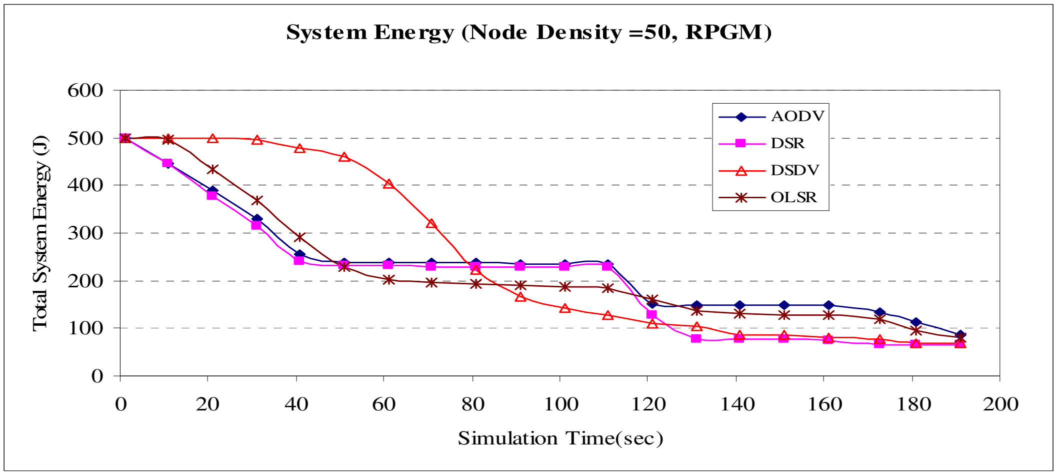


Figure 81. System Energy at Node Density = 50 Nodes

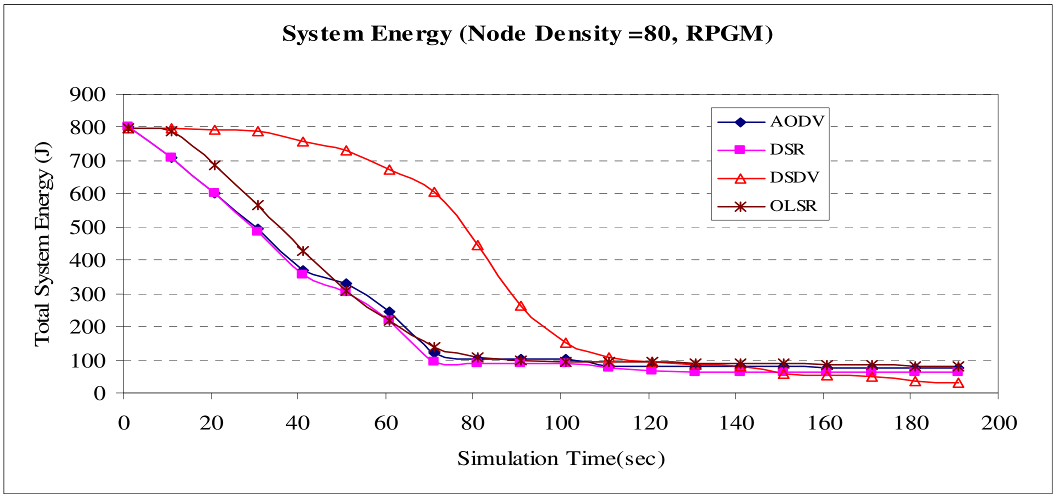


Figure 82. System Energy at Node Density = 80 Nodes

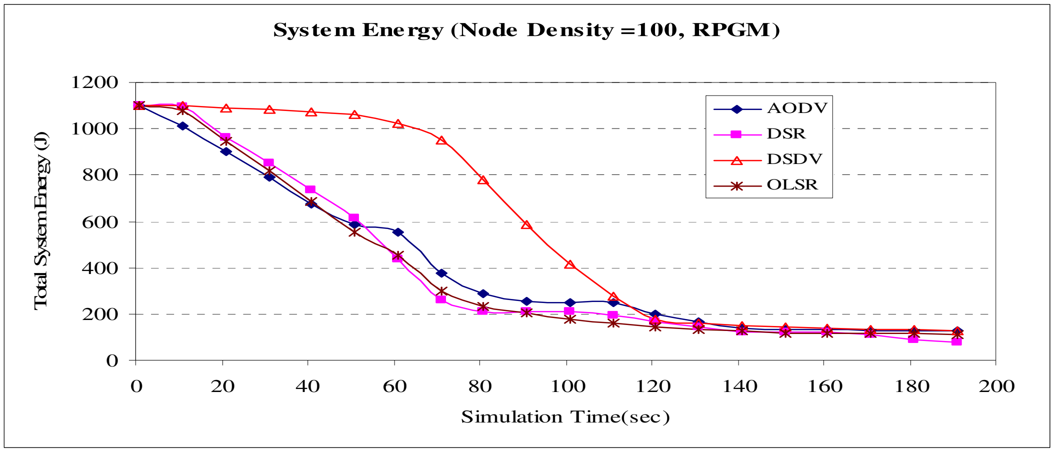


Figure 83. System Energy at Node Density = 100 Nodes

3. Network Loading - Results

In the final simulation for the RPGM model, the goal is to investigate the routing performance aspect when the offered load increases by increasing the average connection load offered by each connection, starting at 20 pkts/sec to 60 pkts/sec. This study of increase in network loading can be used to understand the impact of bandwidth intensive applications such as video streaming. The same parameters are used, such as the type of traffic is CBR and the nodes are pumping data. Table 16 summarizes the parameters.

Simulation Parameters	
Protocols	AODV, DSR, DSDV, OLSR
Simulation Time	200s
# of Nodes	20
Map Size	1000m x 1000m
Speed	10 m/s
Mobility Model	Reference Point Group Mobility (RPGM)
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	20-30-40-50-60 pkts/sec
Pause Time	20-30s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 16. Simulation Parameters for RPGM – Network Loading Variations

Figure 83 illustrates that the average packet delivery ratio of AODV, DSR and OLSR do not differ much as the network loading increases from 20 to 60 pkts/sec. DSDV has the worst performance amongst the four as the network loading increases. Overall, the protocols experience a drop in packet delivery ratio of approximately 40% (DSR and OLSR) from 75% as network loading increases three fold.

Figure 84 shows the routing overheads graph. Both DSDV and OLSR have almost a constant amount of routing overheads despite the network loading. This demonstrates that the routing overheads are independent of the network loading for these two routing protocols. As for AODV and DSR, there is not much change in the overheads as the connection load increase, and the variation is not significant enough to conclude any correlation between the overheads and network loading. However, it is clear that OLSR still generates much more overheads than DSDV, AODV and DSR.

The average network delay graphs in Figure 85 do suggest that AODV and OLSR have the lowest network latency in the RPGM model when network loading increases. However, the network delay actually rises to a maximum point at around network loading of 40 pkts/sec before decreasing as the network loading increases. In this RPGM model, the nodes converges to a central location, and the relative distance between the nodes do decreases as time passes. As such, the overall delay caused by transmission over a shorter range may have a more significant impact in decreasing the average network delay since not too many routing overheads have been generated as loading increases. As seen from the graph, the results also suggest that DSR and DSDV do not adapt as well, in terms of network latency, as the loading increases.

Figures 86 to 90 illustrate the system energy graphs for each network loading. The DSDV nodes consume energy at a relatively slower pace than OLSR, DSR and AODV. However, upon approaching the half way mark in the simulation, the energy level of the system decreases significantly for DSDV as more nodes are sending at the same rate. Whereas in the case of AODV and DSR, once routes have been established, the increase in more connections (more nodes start transmitting as the simulation time increases) or connections load do not greatly impact the overall system energy as lesser routes are generated in the process. Think of AODV and DSR as being more energy-efficient than DSDV and OLSR in this scenario.

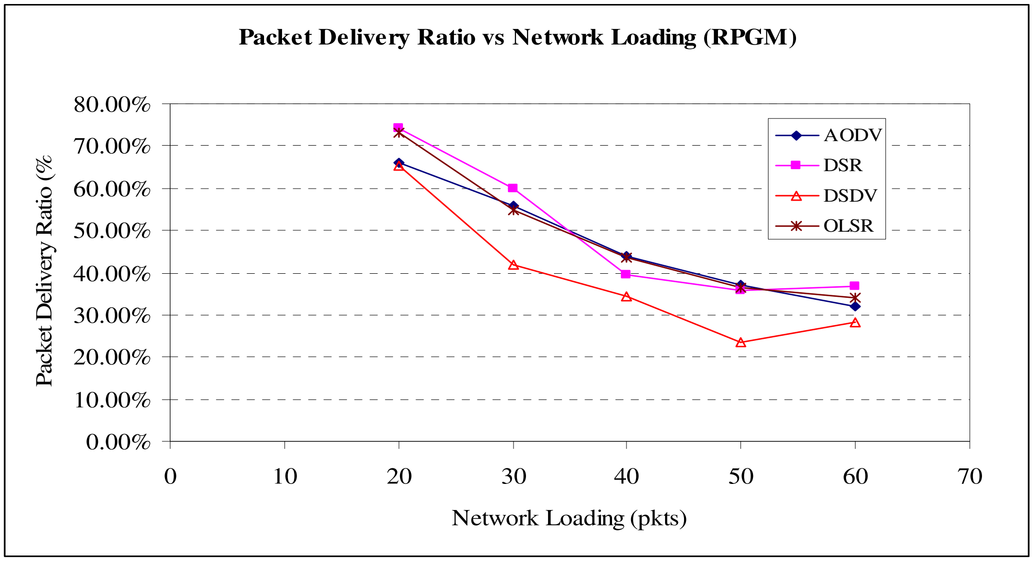


Figure 84. Packet Delivery Ratio with varied Network Loading (20-60 pkts/sec) – RPGM

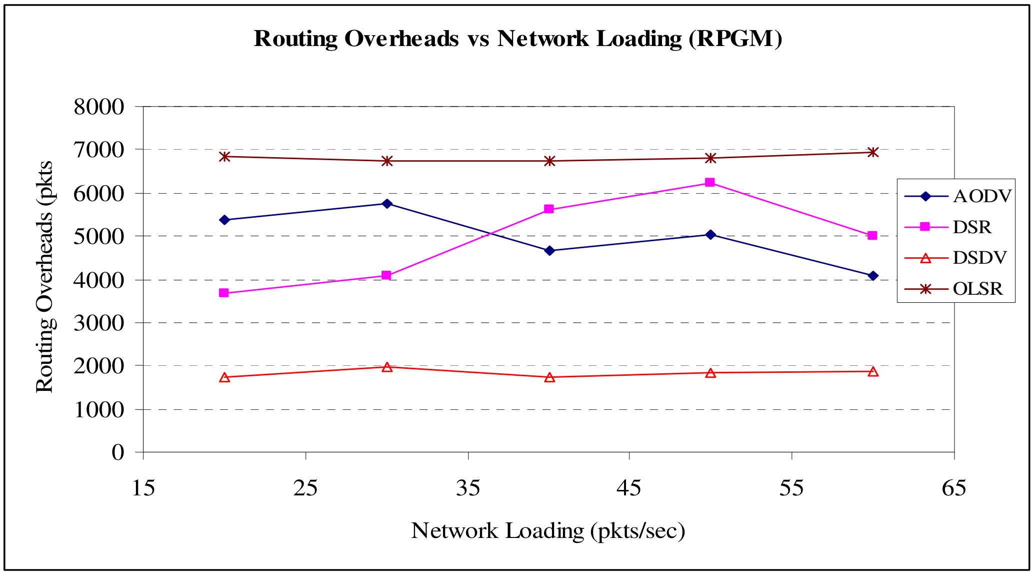


Figure 85. Routing Overhead with varied Network Loading (20-60 pkts/sec) – RPGM

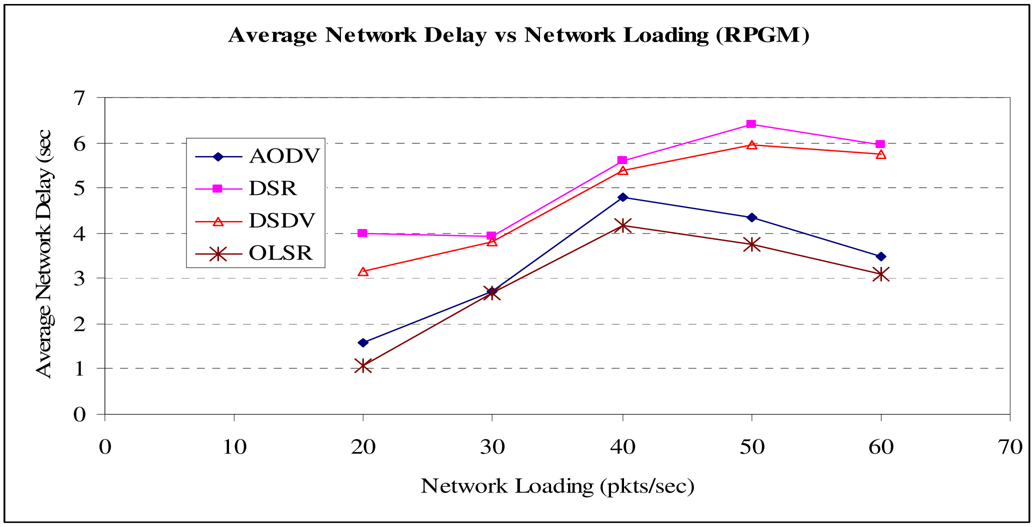


Figure 86. Average Network Delay with varied Network Loading (20-60 pkts/sec) – RPGM

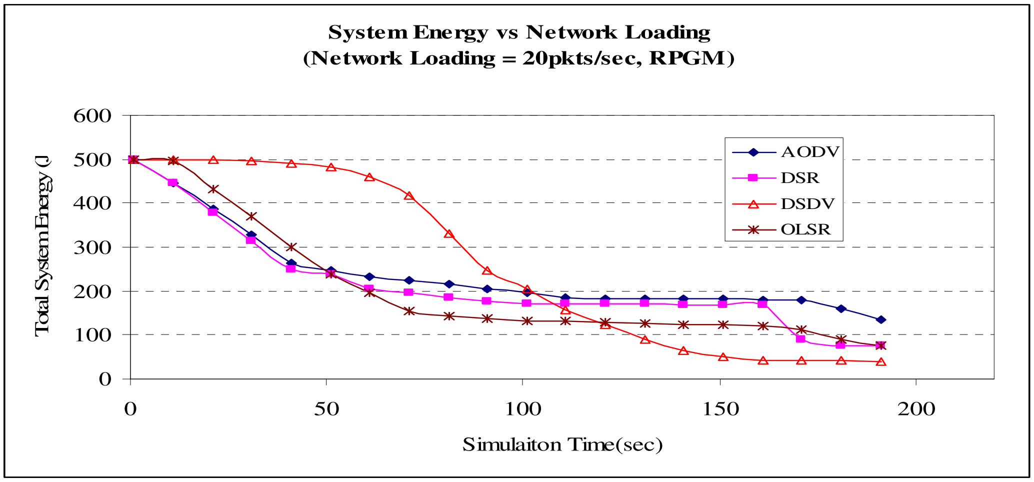


Figure 87. System Energy at Network Loading = 20pkts/sec

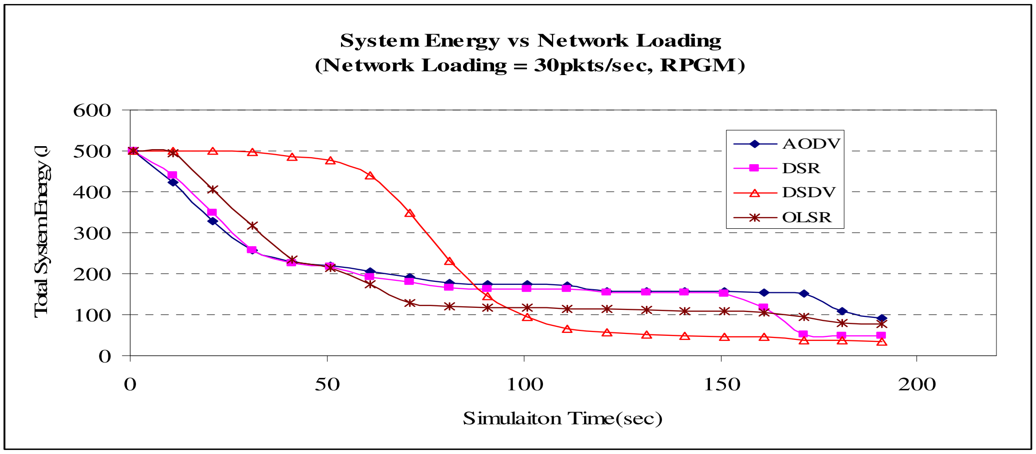


Figure 88. System Energy at Network Loading = 30 pkts/sec

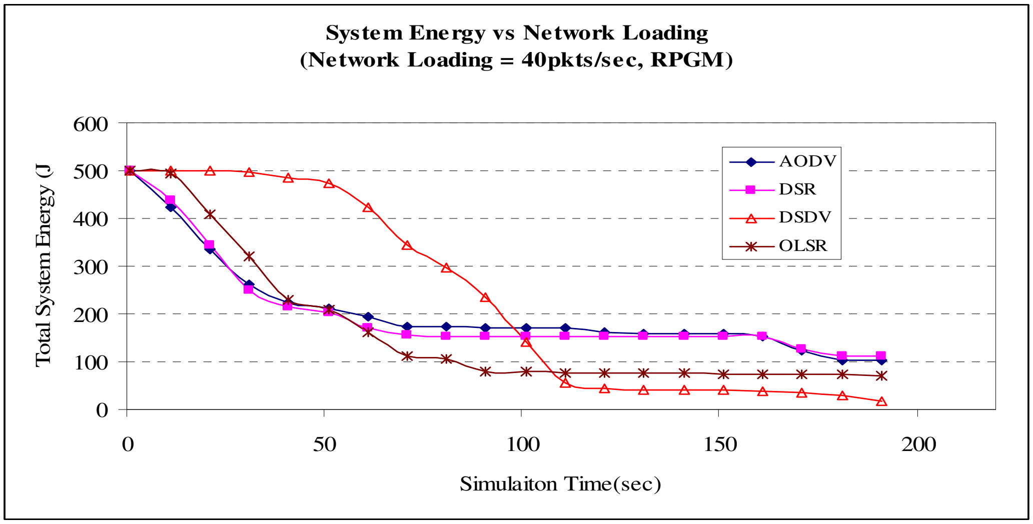


Figure 89. System Energy at Network Loading = 40 pkts/sec

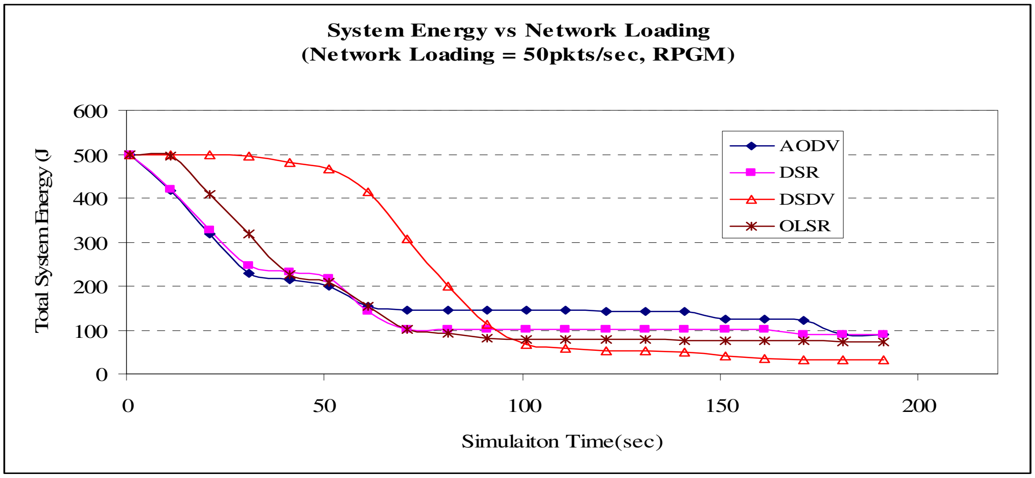


Figure 90. System Energy at Network Loading = 50 pkts/sec

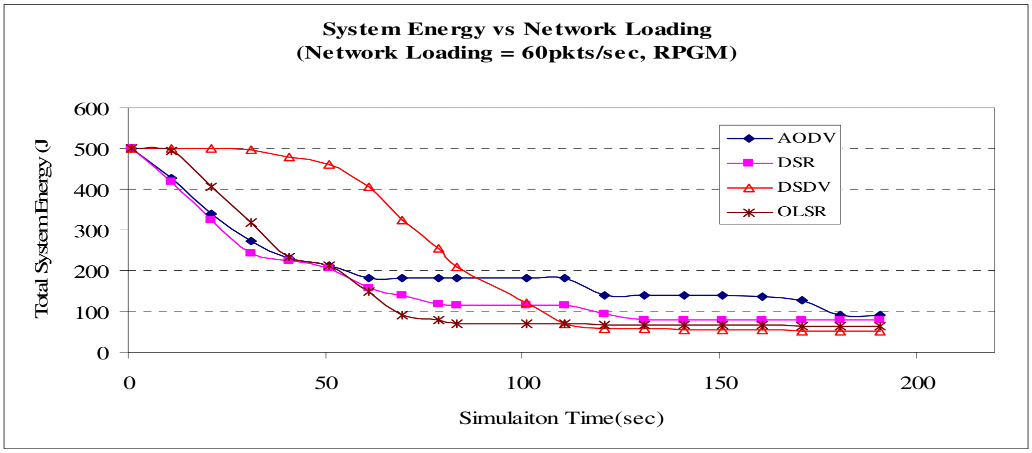


Figure 91. System Energy at Network Loading = 60 pkts/sec

D. OLSR PARAMETER PERFORMANCE

This section is dedicated to the study of the performance of OLSR specifically by tuning its parameters. Chapter III discussed the parameters for which it is possible to adjust for OLSR. The RFC 3626 [RFC3626] suggests default values fpr *Hello Intervals* (Hi) and *Topology Control Intervals* (TCi), which were highlighted in Chapter III, paragraph D. In this simulation experiment, the desire is to investigate whether these are the optimum default values to be used under all circumstances. The Random Waypoint model is used for mobility and due to limited resources and time constraints, it is the only model investigated. Future works will investigate other mobility models.

1. Random Waypoint Model, Speed = 10m/s

This simulation set up considers a Random Waypoint model with nodes moving at speed of 10 m/s with varied topology control intervals and hello intervals, respectively, between 2 to 8 sec and 1 to 10 sec. The default value of Hi and Tci proposed in RFC is 2 sec and 5 sec, respectively. The traffic type is CBR. Table 17 highlights the other parameters of simulation.

Simulation Parameters	
Protocols	OLSR
Simulation Time	200s
# of Nodes	30
Map Size	670m x 670m
Speed	10 m/s
Mobility Model	Random Waypoint
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	20 pkts/sec
Hello Intervals	1 to 10 sec
Topology Control Intervals	2 to 8 sec
Pause Time	2s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 17. Simulation Parameters for OLSR Tweaking, Speed = 10m/s

Figure 91 shows the results of the simulation, which indicates the performance of packet delivery ratio of OLSR for all combinations of Hi (1-10 sec) and TCi (2-8 sec) and Figure 93 illustrates the overheads generated by the routing protocol during the simulation. Figure 94 demonstrates the average network delay experienced and Figure 97 the final system energy states for each different combination of Hi and TCi at the end of the simulation.

From Figure 91 it is possible to observe a general *declining* trend of packet delivery ratio (PDR), for a given Tci value, as the Hi value is varied from 1 to 10 sec. In the case of Tci = 5sec, Figure 92 shows the PDR performance of varying Hi values. As the Hi changes from 1 to 10 sec, the PDR drops from 75.5% to approximately 71%. As such for any given fixed Tci value, the PDR is indirectly proportional to hello intervals, i.e., as Hi increases, PDR decreases. This is logical since the hello interval timing basically represents how "fresh" the routes are. In a shorter hello interval, routes are updated at a more frequent rate, and hence, they are more current. Shorter hello intervals will lead to better packet delivery rates. Likewise, the routing overheads have the same kind of relationship PDR has with hello intervals, at a fixed Tci value. Thus, in order to improve on PDR performance, set a *lower* value of Hi value, for a fixed Tci. However, that would also mean that, for this combination of Hi and Tci, a higher routing overheads occurs.

On the other hand, the network delay graph does not provide a general trending as the variables varied. Thus, it is possible to , from Figure 94, that network delays can be treated as within a *band* of possible values, as the hello intervals increase. Take the case of Tci = 5sec, as Hi increases from 1 to 10 sec, observe that the value of network delay "zig-zags" up and down but stays within the boundary of 1.5 to 3 sec delay. For the default pair of <Hi,Tci> = <2,5>, the delay is roughly 2.5 sec. Thereafter, when Hi is above 2 to 10, all the network delay values are below 2.5 sec, and hence, make it a lower latency network. Therefore, <2,5> is not necessarily optimum in performance from the latency standpoint.

From the final system energy perspective, Figure 96 suggests that there is no clear trend when varying these two variables under study. For <Hi,Tci> = <2,5>, the final system state has about 2J of energy left. There are many combinations that are better off (with higher energy value left) and amongst the best options are <3,7>, <6,7> and <5,6>, all of which are close to 4.5J of energy left. Looking then at Figure 96, <3,7> and <6,7> both have the same average network delay of 3 sec while <5,6> has an average delay of 1.5 sec (see Figure 94), which is half in value. As such, <5,6> can be chosen over <3,7> and <6,7> as the *local optimum* point for OLSR, bearing in mind, from a specific purpose and in this case, maximizing system energy and minimizing network delay. The PDR is indifferent in this particular case.

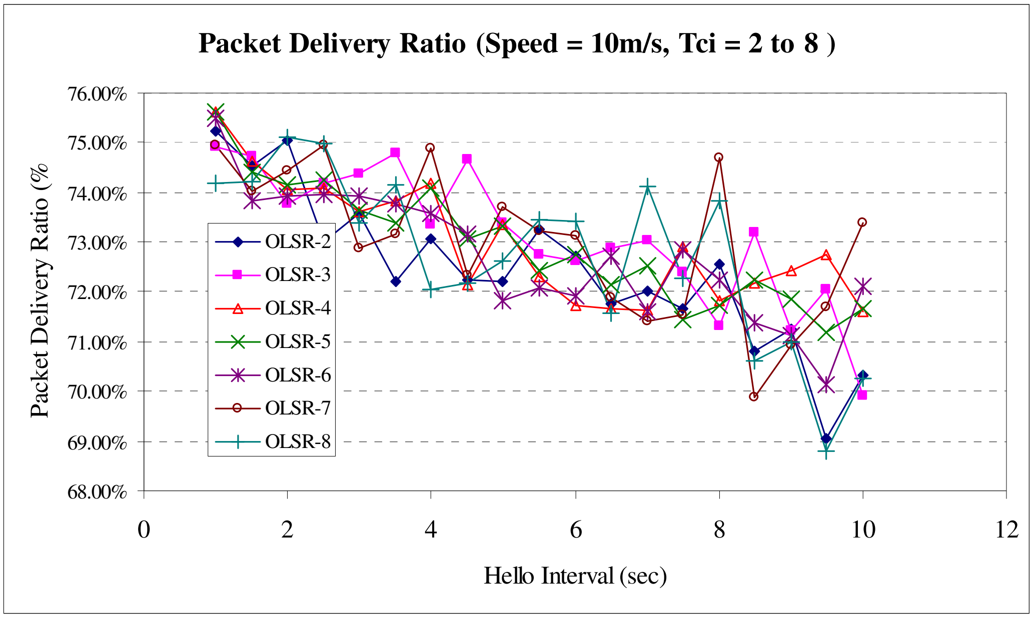


Figure 92. Packet Delivery Ratio with varied Hello Intervals and Topology Control Intervals, Speed = 10m/s – Random Waypoint

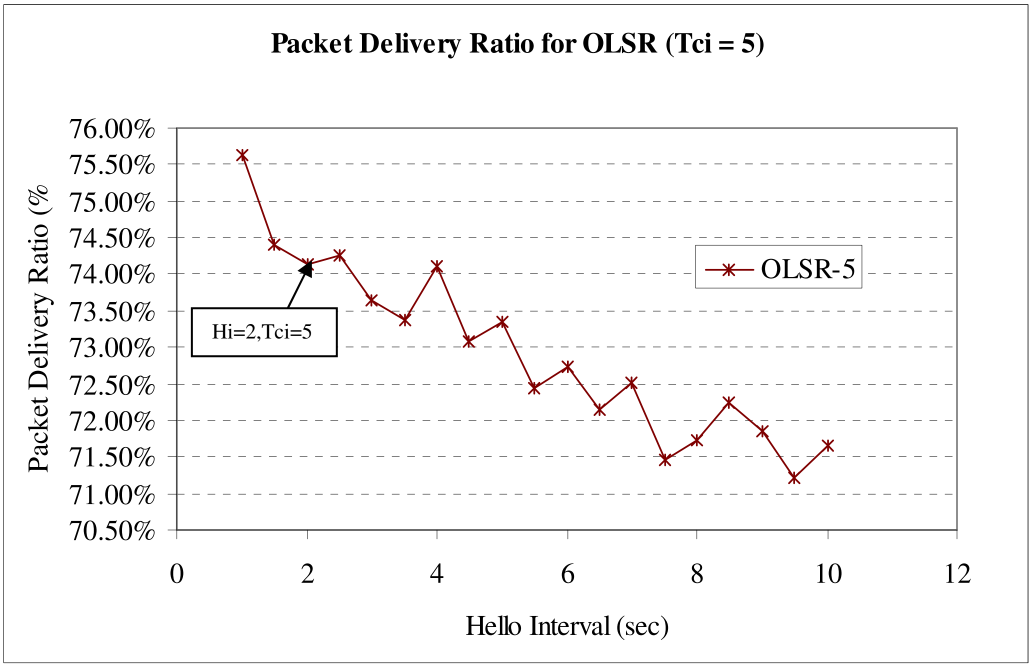


Figure 93. Packet Delivery Ratio with varied Hello Intervals and Topology Control Interval = 5 sec at Speed = 10m/s – Random Waypoint

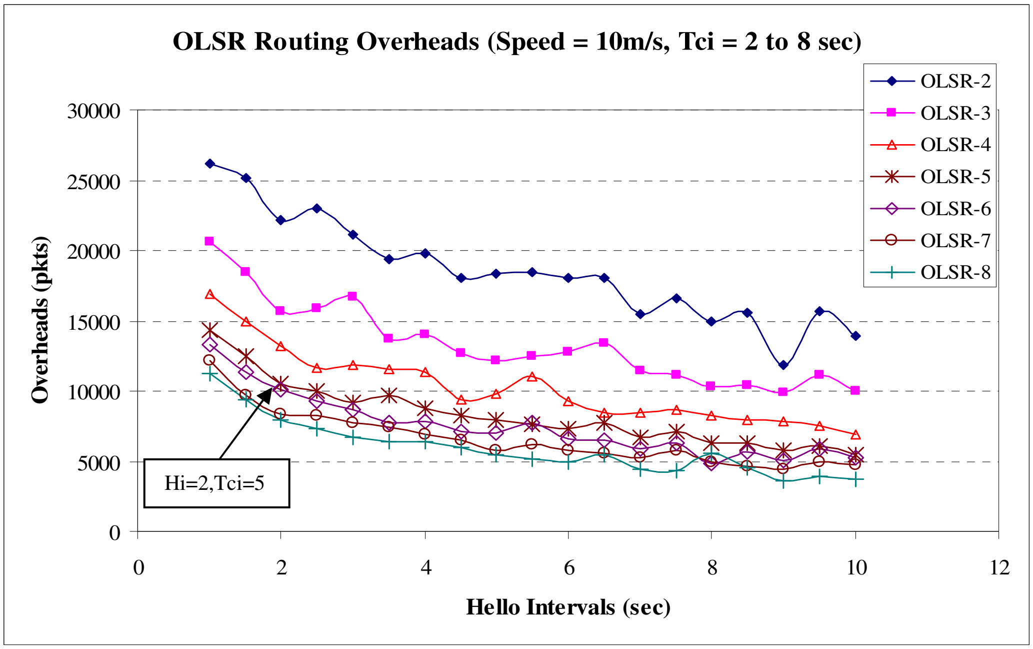


Figure 94. Routing Overheads with varied Hello Intervals and Topology Control Intervals, Speed = 10m/s – Random Waypoint

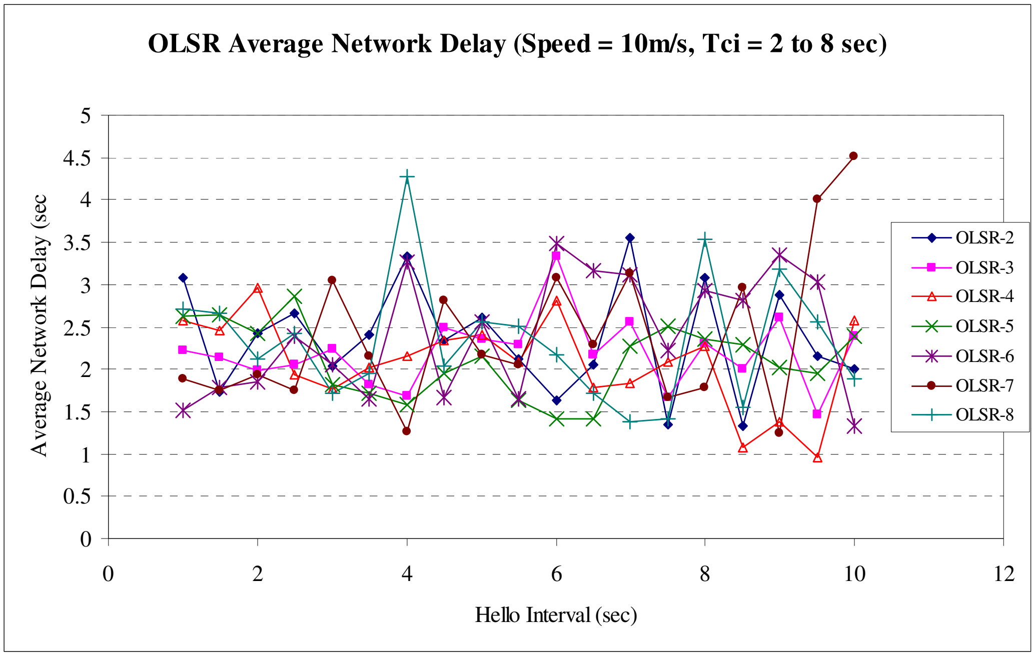


Figure 95. Average Network Delay with varied Hello Intervals and Topology Control Intervals, Speed = 10m/s – Random Waypoint

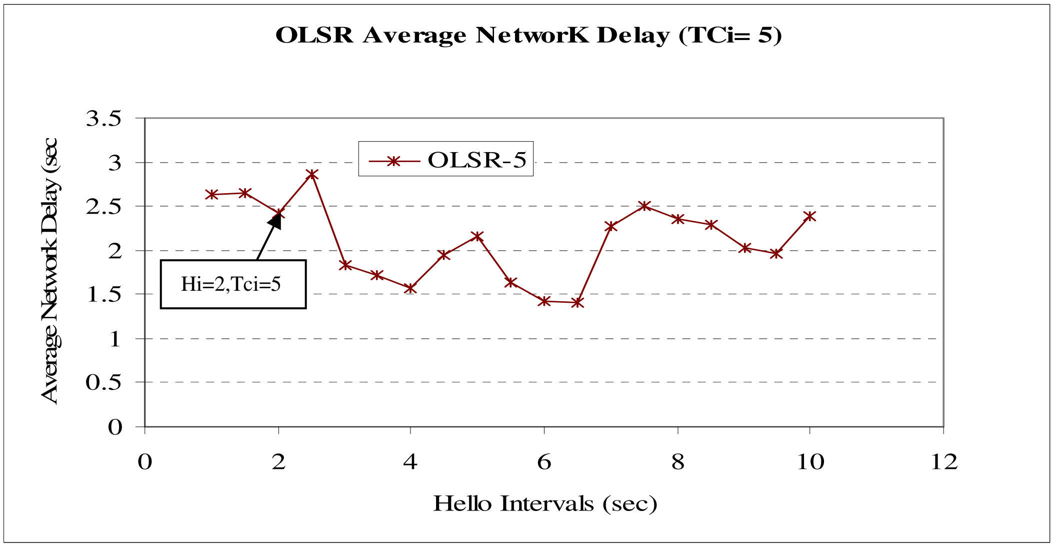


Figure 96. Average Network Delay with varied Hello Intervals and Topology Control Interval set at 5 sec, Speed = 10m/s – Random Waypoint

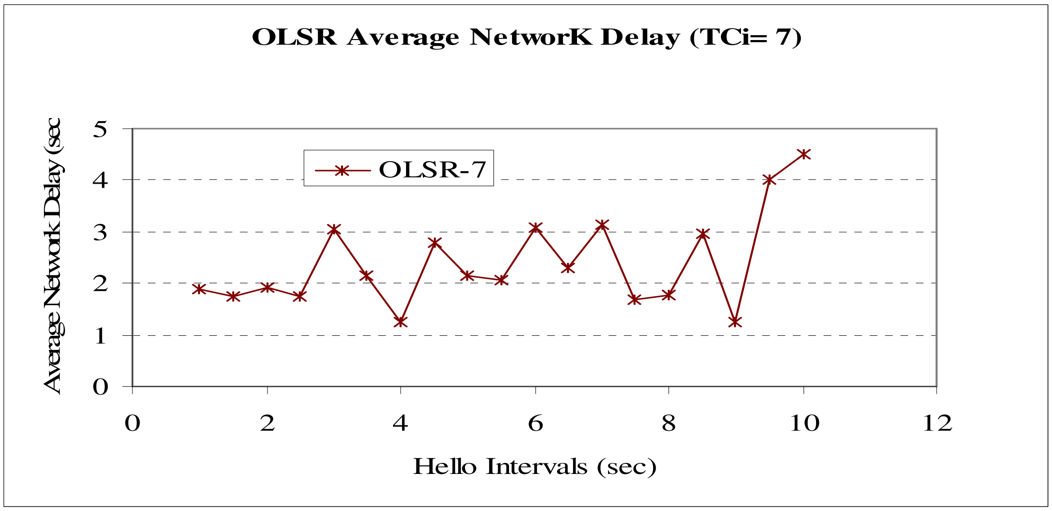


Figure 97. Average Network Delay with varied Hello Intervals and Topology Control Interval set at 7 sec, Speed = 10m/s – Random Waypoint

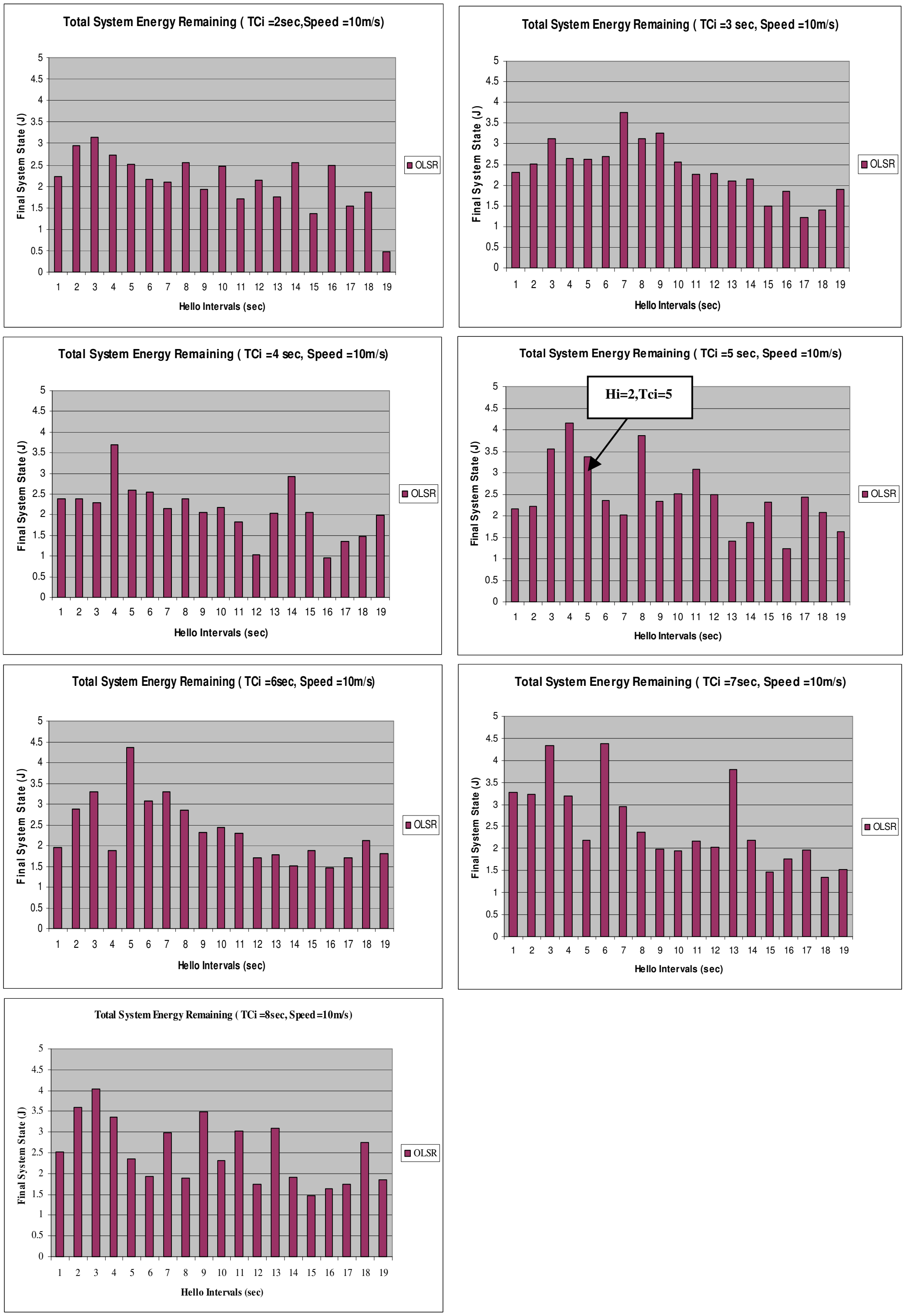


Figure 98. System Energy with varied Hello Intervals and Topology Control Intervals, Speed = 10m/s – Random Waypoint

2. Random Waypoint, Speed = 25m/s

This simulation set up considers a Random Waypoint model with nodes moving at a faster speed of 25 m/s. Again, the topology control intervals and hello intervals were varied, respectively, between 2 to 8 sec and 1 to 10 sec. The traffic type is CBR. Table 18 highlights the other parameters of simulation.

Simulation Parameters	
Protocols	OLSR
Simulation Time	200s
# of Nodes	30
Map Size	670m x 670m
Speed	25 m/s
Mobility Model	Random Waypoint
Traffic Type	Constant Bit Rate (CBR)
Packet Size	512 bytes
Connection Rate	20 pkts/sec
Hello Intervals	1 to 10 sec
Topology Control Intervals	2 to 8 sec
Pause Time	2s
Node Energy	10 Joules per node
Receive Power	300 mW
Transmit Power	800 mW
# of connections	5

Table 18. Simulation Parameters for OLSR Tweaking, Speed = 25m/s

Figure 98 indicates the packet delivery ratio for OLSR when performing the simulation at speed 25m/s. Generally, the similar trend exists as in the case of 10m/s, i.e., the packet delivery ratio drops from 74% to around 70% as the intervals for Hello and Topology Control updates increase. Figure 99 shows the specific case for Tci = 5sec.

Figure 100 presents the overall routing overheads. As the interval increases, a declining rate of the routing overheads can be seen, Also, at a higher mobility speed (25m/s), there is a much lower routing overheads than the lower mobility case with the same hello interval and topology control interval used for the simulation.

Figures 101 and 102 seem to indicate no specific trend between the average network delay when varying the hello and topology control intervals. The network delay is “bounded” within an upper and lower bound value of 4 and 1 sec, with a majority of the plotted points falling within them. However, the network delay seems to “diverge” – the delay has higher and lower values as the hello interval increases and the system response is seemingly quite unstable. This may result from increasing the waiting interval for updates. In a highly dynamic mobile environment, many changes can take place within this refresh period, and thus, making the delay more erratic as nodes change routes more often.

Figure 103 shows the final system energy state at the end of each simulation as the hello and topology control intervals are changed. It is interesting to observe that for a fixed Tci = 5 sec, generally, many hello intervals have better final system energy than the majority of other possible combinations. Combination pair <5,7> has the most energy saving system with the final system energy state at the highest. Looking at the packet delivery graph and the average network delay graph for Tci = 5 sec, note that for hello intervals equal to 7 sec, approximately 72.5% and 1.25 sec is achieved. The maximum packet delivery rate is close to 75% and the lowest latency is about 1 sec, as such, <5,7> can be considered to be a good balance point between generating too much routing overhead and having enough routing traffic to ensure a good packet delivery rate and achieve low network latency.

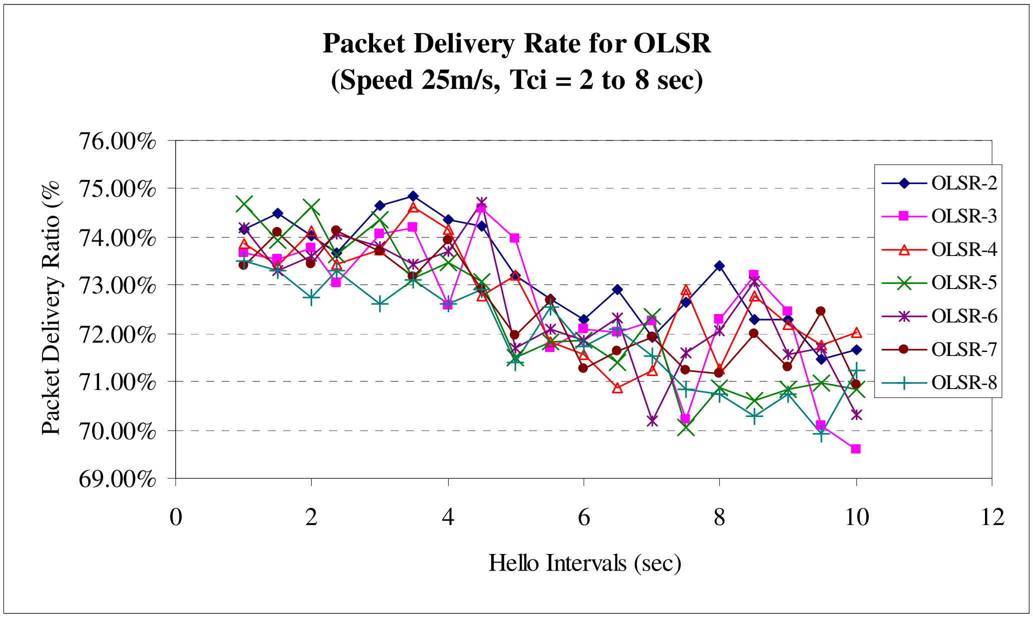


Figure 99. Packet Delivery Ratio with varied Hello Intervals and Topology Control Intervals, Speed = 25m/s – Random Waypoint

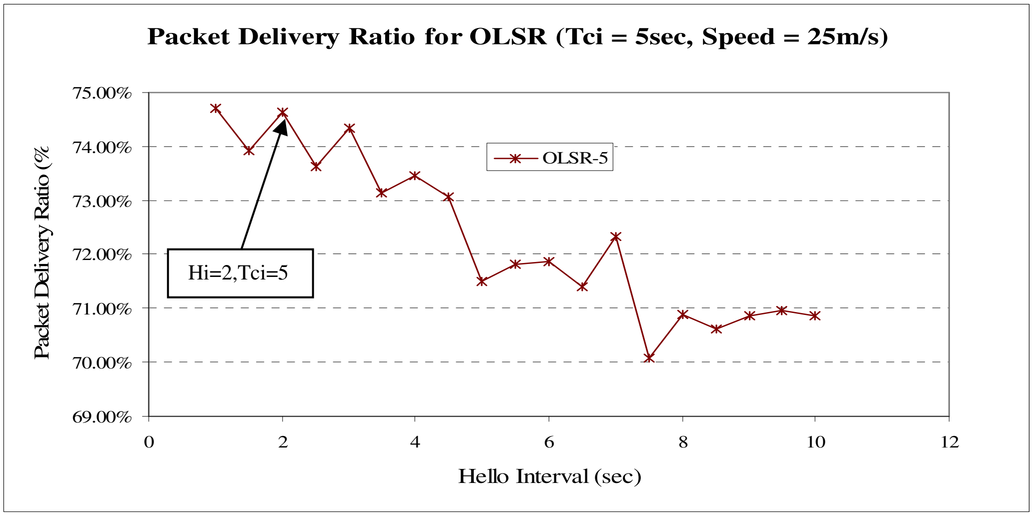


Figure 100. Packet Delivery Ratio with varied Hello Intervals and Topology Control Interval = 5 sec at Speed = 25m/s – Random Waypoint

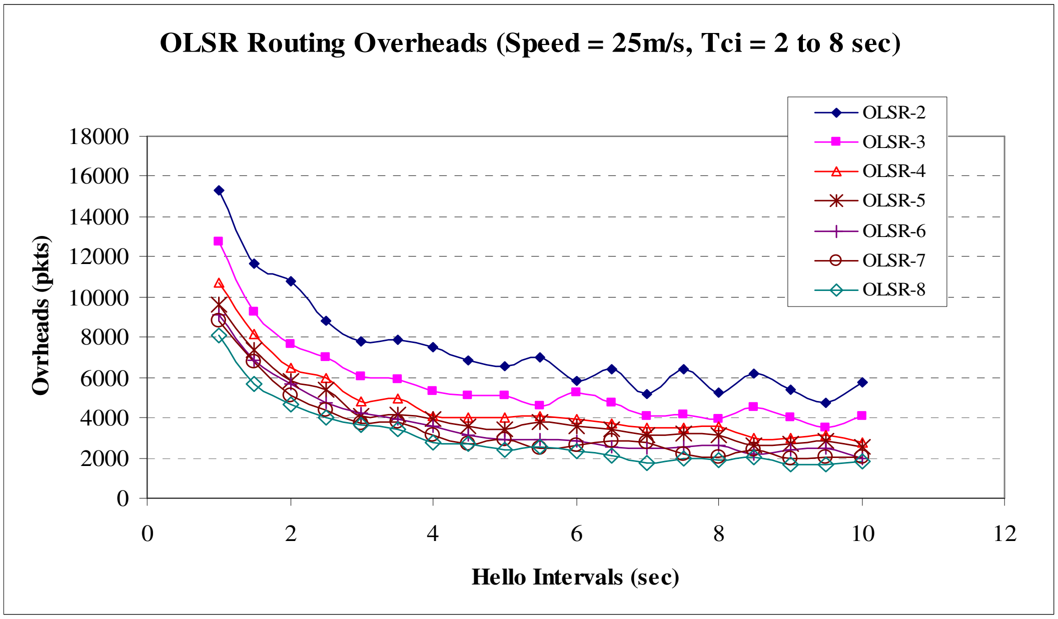


Figure 101. Routing Overheads with varied Hello Intervals and Topology Control Intervals, Speed = 25m/s – Random Waypoint

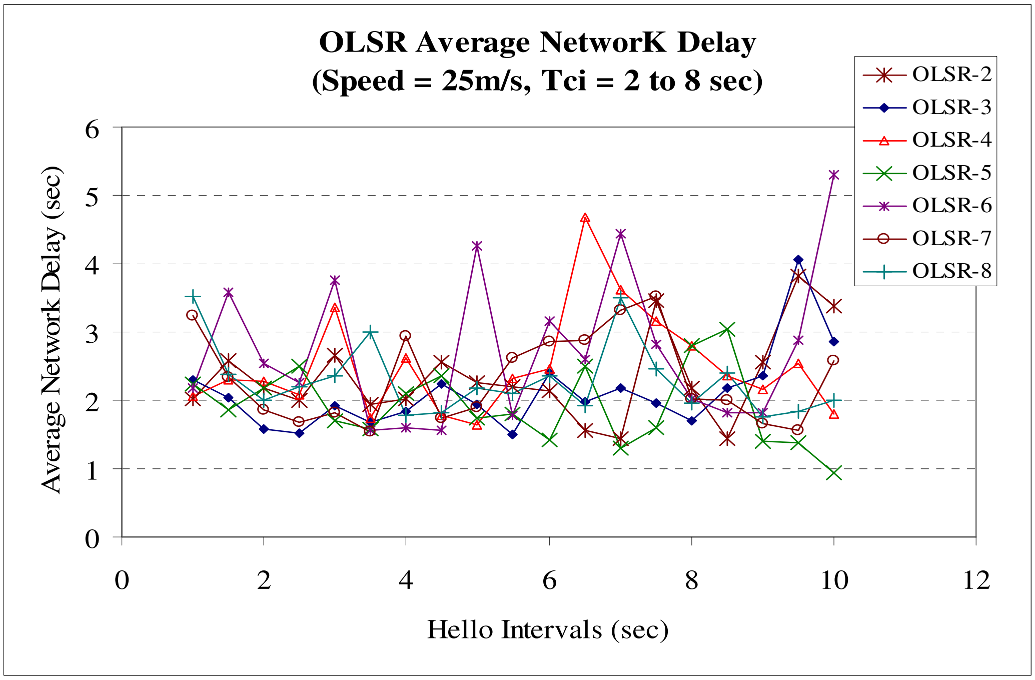


Figure 102. Average Network Delay with varied Hello Intervals and Topology Control Intervals, Speed = 25m/s– Random Waypoint

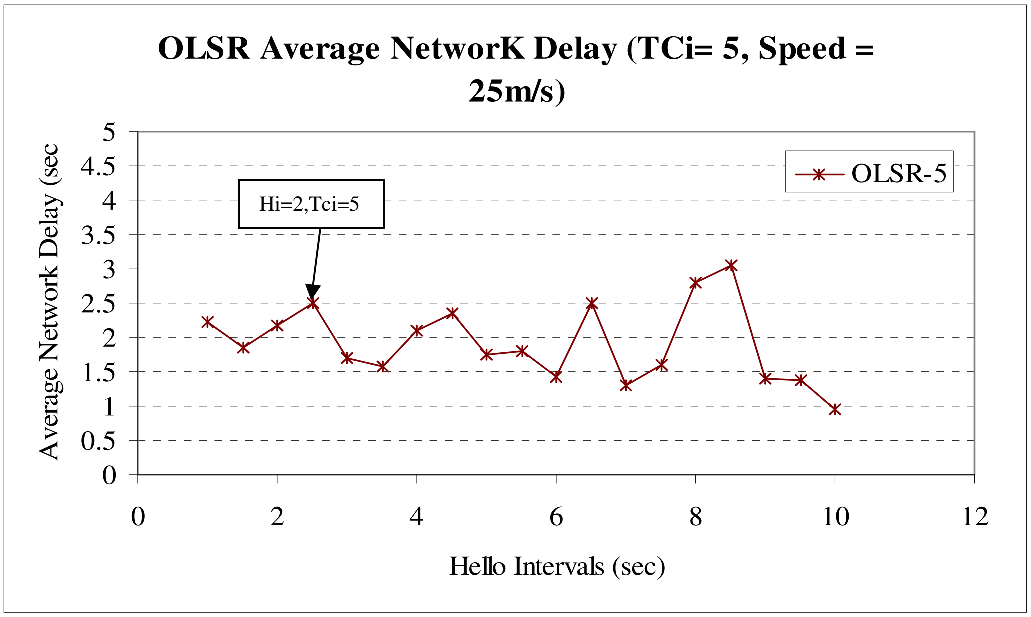


Figure 103. Average Network Delay with varied Hello Intervals and Topology Control Interval set at 5 sec, Speed = 25m/s – Random Waypoint

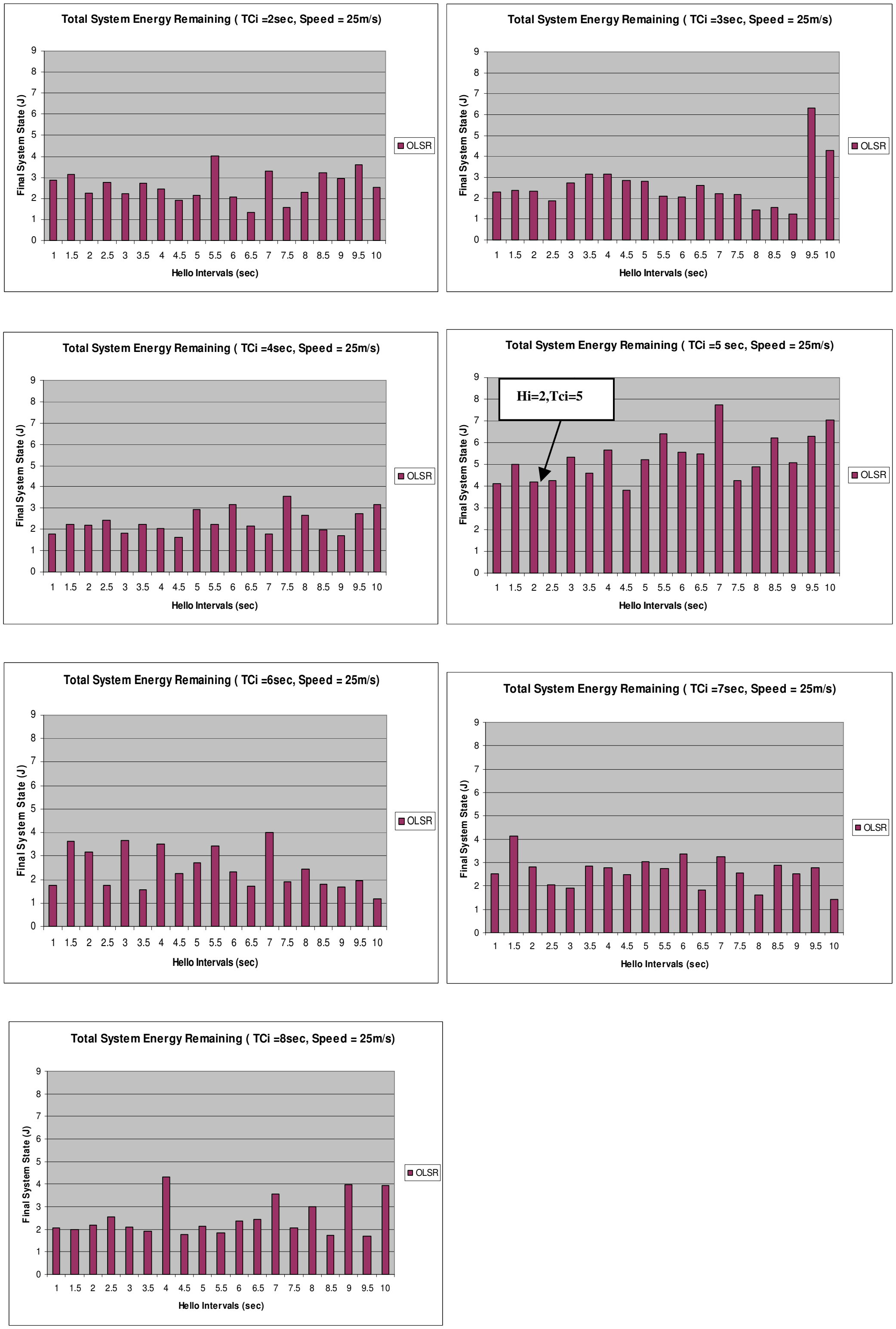


Figure 104. System Energy with varied Hello Intervals and Topology Control Intervals, Speed = 25m/s – Random Waypoint

VI. CONCLUSION, RECOMMENDATIONS AND FUTURE WORKS

A. CONCLUSION AND RECOMMENDATIONS

This thesis studies four proposed ad hoc routing protocols, namely, AODV, DSR, DSDV and OLSR by means of simulation in network environments of 670m x 670m and 1000m x 1000m using varied mobility speeds, offered network loadings as well as node densities, in order to determine the network performance metrics such as overall packet delivery ratio, average network latency, routing overheads as well as total system energy consumption. Different mobility models are explored. The most commonly studied mobility models such as the Random waypoint were used, as well as the less researched mobility models in ad hoc networking such as the Manhattan Grid and the Reference Point Group mobility models. The output of the mobility model files was modified to suit a realistic situation for military maneuvers and this situation was applied in hopes of better understanding the impact of these routing protocols on overall system performance.

In addition, the second part of the simulation concentrated on OLSR, and its two most important parameters: *hello interval* and *topology control interval*. Their values were varied in order to understand whether the default recommended values can achieve network optimum performance and used the same type of metrics for evaluation as in the first experiment.

The conclusions drawn from the simulation results obtained are as follows:

- OLSR has emerged as a good compromise (cf. Figure 16/18 and 24/26) between high packet delivery ratio and network latency. It has consistent performance and in some cases (cf. Figure 47/62/77/86), better simulation results than other routing protocols. It has comparable packet delivery ratio as on-demand routing protocols and yet maintains relatively low network latency. This is important for delay-sensitive applications. The trade-off for using OLSR is that there may be a considerable amount of routing overheads.
- *Recommendation.* As our models have suggested, currently, OLSR can be a suitable ad hoc routing protocol for military operations, especially in the case where delay-sensitive applications are deployed. Whereas in cases where delay is not a major concern, on-demand protocols are good alternatives as they present themselves as more bandwidth-efficient and more power efficient in the wireless networks. Keep in mind that these

observations are valid only for the mobility and traffic models considered in this thesis. Additional studies should be conducted for application scenarios where the mobility and traffic generation patterns deviate from our models.

- The performance of the OLSR protocol is very sensitive to the values of hello and topology control interval parameters under the Random Waypoint mobility model. The proposed default values (2 and 5 seconds) for these parameters have often not been the optimal choice and in some cases the performance degradation is significant.
- *Recommendation.* We recommend conducting further studies to determine if it is possible to dreive the optimal or near-optimal OLSR timing parameters from specific network and application settings. Meanwhile, one may figure out the right OLSR parameters empirically, i.e., by trying out few combination pairs of hello and topology control intervals in computer simulations. According to one's specific network and needs, one may choose the combination that is best suited to one's objectives. We observe such results vary according to node mobility as well.

The detailed observations for the first part of our study are:

- For Random Waypoint model, in dense (node-congested) network, on demand protocols perform better than table-driven types (cf. Figure 24/25/26). System energy levels do not defer much.
- For Random Waypoint model, as we increase the overall network loading by increasing the connection load between pairs of nodes in the system, we observed that the packet delivery ratio drops significantly by almost 30% when the loading is increase by five folds (cf. Figure 32). In this scenario, OLSR conserves system energy better and DSDV has the lowest network latency (cf. Figure 34/35/36/37/38/39). When we increase the number of available connections during the simulation entire duration instead of the node's sending rate, we observe that on-demand protocols have better packet delivery ratio but worse network latency than table-driven protocols (cf. Figure 40/41/42). The gaps, for network latency, diverges significantly (almost three times) as the number of connections increases.
- One striking difference in Manhattan Grid model simulation, for varying speed of nodes, is that the AODV protocol has the worst performance for packet delivery ratio despite being having the best performance results in Random Waypoint model (cf. Figure 47). Moreover, the network latency is the lowest for table-driven protocols in Manhattan Grid models, thereby making them better candidates for high speed nodes in Manhattan Grid-like scenarios. The energy consumption performance indicates a relatively indifference in performance for the four with DSDV having the best performance (cf. Figure 48-54).

- In densely networks of the Manhattan Grid models, the packet delivery ratio performances for AODV, DSR and OLSR are relatively near to one and another. DSR, however, has very bad results in network latency making AODV and OLSR has favorable choices in more dense networks (cf. Figure 55-61).
- There is seemingly no difference in the choice of routing protocols for systems having increasing network load when simulated in a Manhattan Grid model. All protocols have almost similar packet delivery ratio performance (cf. Figure 62). However, DSDV has divergent average network delays duration as the network loading increases (cf. Figure 64). The increase can be as large as five times as we doubled the connection load.
- In the Reference Point Group Mobility model, as we model the differences in node velocity, we observe that AODV, DSR and OLSR have much better packet delivery ratio performance than DSDV (cf. Figure 70). However, OLSR and AODV have the added advantage of lower network latency than DSR (cf. Figure 72).
- OLSR emerges as the best candidate for routing protocol when we operate in a denser network having the RPGM model type mobility. Moreover, OLSR has the lowest network latency (cf. Figure 79), making it the ideal candidate to be deploy in such scenario. One possible setback for OLSR is the relative higher routing overheads (still lower than AODV here, in any case) and its poorer energy consumption rate. Overall, OLSR is still a better candidate (cf. Figure 77-83).
- For increases in network loading, routing by AODV, OLSR or DSR have no significant difference for the RPGM mobility model. But at heavier loading, we have OLSR having the lowest network latency although AODV do not differ much from OLSR (cf. Figure 86). In terms of energy performance, AODV is more energy-efficient than OLSR and thereby consumes less overall system energy. As such for circumstance such as military operation using energy-deprive sensory, AODV may emerge as a better candidate (cf. Figure 84-91).

In the second part of the simulation, we study the behavior and performance of OLSR protocol when we vary the Hello Interval and Topology Control Interval. We note that:

- In both mobility speed scenarios (10m/s and 25m/s), as we increase the hello interval and topology control interval, we have generally obtained lower packet delivery rate for the system (cf. Figure 92/93/99/100).
- The routing overhead graphs show that there is a decrease in the overhead as we increases the hello and topology control intervals (cf. 94/101). The relative drop of the overhead, in percentage terms, remains the same

although the actual count of overhead drop difference is lower as we increase the hello and topology control intervals.

- The change in the average network delay is unpredictable as we vary the hello and topology control intervals. The delay graphs show a “zig-zag” pattern over different hello and topology control interval combiniations (cf. Figure 95/96/102/103). As the hello interval gets larger, this “zig-zag” range seems to diverge more.

B. FUTURE WORKS

Our work here is far from answering all the questions related to mobile ad hoc routing protocols. The analysis of routing protocols for ad hoc networking has been expanded by considering different mobility models as well as tweaking the parameters within the protocols. However, we will need to address specific limitations of this thesis. They include,

- Adding light load traffic generation (e.g., exponential connection time with random source destination pair). In this case, we are able to observe the performance of routing protocols whereby the control traffic overheads dominate the energy consumption, instead of the data load.
- Analyzing the route lengths (hop counts) and understanding who are the major contributors to the overall delay of data transfer. Delay can be dissected to analyse the different contributors to the overall system-to-system connectivity.
- OLSR is a relatively new and under-tested protocol. Other considerations for testing its performance include other mobility models when tweaking OLSR parameters.

APPENDIX A. SAMPLE TCL SCRIPT FILE

The sample TCL script provided here is the configuration file for simulation examples. The parameters of the NS2 simulation is varied and monitored to execute different sets of experimentations successfully.

```
# ===================================================================
# A simple example for wireless simulation
# 20 nodes using AODV as Routing Protocol
# Author : KT Lee, klee@nps.edu 2004
#====================================================================
#====================================================================
# Define options
#====================================================================
set opt(chan)           Channel/WirelessChannel
set opt(prop)           Propagation/TwoRayGround
set opt(netif)          Phy/WirelessPhy
set opt(mac)            Mac/802_11
set opt(ifq)            Queue/DropTail/PriQueue
set opt(ll)             LL
set opt(ant)            Antenna/OmniAntenna
set opt(x)              1000   ;# X dimension of the topography
set opt(y)              1000  ;# Y dimension of the topography
set opt(ifqlen)         50          ;#packet in ifq
set opt(seed)           0.0
set opt(tr)             aodv20r.tr    ;# trace file
set opt(nam)            aodv20r.nam   ;# nam trace file
set opt(adhocRouting)   AODV ;#routing protocol
set opt(nn)             50             ;# how many nodes are simulated
set opt(cp)             “cbr-20-c10r20”  ;# Traffif generation file
set opt(sc)             “MH-20-s5.ns_movements”  ;# Scenario Mobility
file
set opt(stop)           200.0             ;# simulation time
#set opt(energyModel)   EnergyModel       ;# To be activated when
evalutating energy
#set opt(initialenergy) 10                ;# performance of Routing
protocols
#set opt(p_rx)          0.3
#set opt(p_tx)          0.8
#set opt(p_idle)        0.0
# ===================================================================
# Other default settings
# ===================================================================
LL set mindelay_              50us
LL set delay_                 25us
LL set bandwidth_             0     ;# not used
Agent/Null set sport_         0
Agent/Null set dport_         0
Agent/CBR set sport_          0
Agent/CBR set dport_          0
Agent/TCPSink set sport_      0
Agent/TCPSink set dport_      0
Agent/TCP set sport_          0
```

```
Agent/TCP set dport_          0
Agent/TCP set packetSize_     512
Queue/DropTail/PriQueue       set Prefer_Routing_Protocols     1

#----------------------------------------------------------------------
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5 meters above
# it
#----------------------------------------------------------------------
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0
#----------------------------------------------------------------------
# Initialize the SharedMedia interface with parameters to make
# it work like the 914MHz Lucent WaveLAN DSSS radio interface
#----------------------------------------------------------------------
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.2818
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0
# ====================================================================
# Main Program
# ====================================================================
#=====================================================================
#READ INPUTS FROM COMMAND LINE
#=====================================================================
#set opt(tr) [lindex $argv 0]
#puts “$opt(tr) trace file is loaded ... “
#set opt(nam) [lindex $argv 1]
#puts “$opt(nam) nam file is loaded ... “
#set opt(cp) [lindex $argv 2]
#puts “$opt(cp) scenario is loaded ... “
#et opt(sc) [lindex $argv 3]
#puts “$opt(sc) scenario is loaded ... “
#set opt(nn) [lindex $argv 4]
#puts “$opt(nn) nodes are loaded ... “
#=====================================================================
# Initialize Global Variables
#=====================================================================

# create simulator instance
set ns_           [new Simulator]
#=====================================================================
# set wireless channel, radio-model and topography objects
#=====================================================================
#set wchan  [new $opt(chan)]
#set wprop  [new $opt(prop)]
set wtopo   [new Topography]
#=====================================================================
# create trace object for ns and nam
#=====================================================================
set tracefd [open $opt(tr) w]
set namtrace    [open $opt(nam) w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace 500 500
#=====================================================================
# use new trace file format
#=====================================================================
#$ns_ use-newtrace
#=====================================================================
# define topology
#=====================================================================
$wtopo load_flatgrid $opt(x) $opt(y)
#$wprop topography $wtopo
#=====================================================================
# Create God
#=====================================================================
set god_ [create-god $opt(nn)]
#=====================================================================
#global node setting -define how wireless/mobile node should be created
#=====================================================================
$ns_ node-config -adhocRouting $opt(adhocRouting) \
             -llType $opt(ll) \
             -macType $opt(mac) \
             -ifqType $opt(ifq) \
             -ifqLen $opt(ifqlen) \
             -antType $opt(ant) \
             -propType $opt(prop) \
             -phyType $opt(netif) \
             -channelType $opt(chan) \
             -topoInstance $wtopo \
             -agentTrace ON \
             -routerTrace ON \
             -macTrace ON  \
             -MovementTrace ON
#=====================================================================
# If energy logging is required, these are the parameters to be
uncommented
# for mobile node logging
#=====================================================================
             #-energyModel $opt(energyModel) \
             #-initialEnergy 10 \
             #-rxPower $opt(p_rx) \
             #-txPower $opt(p_tx)
#=====================================================================
#  Create the specified number of nodes [$opt(nn)] and “attach” them
#  to the channel.
#=====================================================================
for {set i 0} {$i < $opt(nn) } {incr i} {
      set node_($i) [$ns_ node]
      $node_($i) random-motion 0          ;# disable random motion
#     $node_($i) topography $wtopo
}
#=====================================================================
# Specifics to OLSR - Here to specify the parameters for OLSR
#=====================================================================
if {$opt(adhocRouting) == “ProtolibManetKernel”} {
    for {set i 0} {$i < $opt(nn) } {incr i} {
      set p($i) [new Agent/NrlolsrAgent]
      $ns_ attach-agent $node_($i) $p($i)
      $ns_ at 0.0 "$p($i) startup -tcj .75 -hj .5 -tci 2.5 -hi .5 -d 8
-l /tmp/olsr.log"
      [$node_($i) set ragent_] attach-manet $p($i)
      $p($i) attach-protolibManetKernel [$node_($i) set ragent_]
    }
}
#===========================================================================
# Define node movement model
#===========================================================================
puts "Loading connection pattern..."
source $opt(cp)
#===========================================================================
# Define traffic model
#===========================================================================
puts "Loading scenario file..."
source $opt(sc)
# Define node initial position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {

    # 20 defines the node size in nam, must adjust it according to your
scenario
    # The function must be called after mobility model is defined

    $ns_ initial_node_pos $node_($i) 35
}
#===========================================================================
# Tell nodes when the simulation ends
#===========================================================================
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
#===========================================================================
# tell nam the simulation stop time
#===========================================================================
$ns_ at  $opt(stop)     "$ns_ nam-end-wireless $opt(stop)"
$ns_ at  $opt(stop).000000001 "puts \"NS EXITING...\" ; $ns_ halt"
puts "Starting Simulation..."
$ns_ run
#===========================================================================
# END OF SIMULATION SCRIPT
#===========================================================================
```

APPENDIX B. SAMPLE TRAFFIC FILE

```
#=======================================================================
# nodes: 50, max conn: 5, send rate: 0.050000000000000003, seed: 1.0
#=======================================================================
#
# 1 connecting to 17 at time 2.5568388786897245
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(17) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.050000000000000003
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 “$cbr_(0) start”
#
# 12 connecting to 25 at time 56.333118917575632
#
set udp_(1) [new Agent/UDP]
$ns_ attach-agent $node_(12) $udp_(1)
set null_(1) [new Agent/Null]
$ns_ attach-agent $node_(25) $null_(1)
set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) set packetSize_ 512
$cbr_(1) set interval_ 0.050000000000000003
$cbr_(1) set random_ 1
$cbr_(1) set maxpkts_ 10000
$cbr_(1) attach-agent $udp_(1)
$ns_ connect $udp_(1) $null_(1)
$ns_ at 56.333118917575632 “$cbr_(1) start”
#
# 4 connecting to 16 at time 146.96568928983328
#
set udp_(2) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(2)
set null_(2) [new Agent/Null]
$ns_ attach-agent $node_(16) $null_(2)
set cbr_(2) [new Application/Traffic/CBR]
$cbr_(2) set packetSize_ 512
$cbr_(2) set interval_ 0.050000000000000003
$cbr_(2) set random_ 1
$cbr_(2) set maxpkts_ 10000
$cbr_(2) attach-agent $udp_(2)
$ns_ connect $udp_(2) $null_(2)
$ns_ at 146.96568928983328 “$cbr_(2) start”
#
# 3 connecting to 34 at time 55.634230382570173
#
set udp_(3) [new Agent/UDP]
$ns_ attach-agent $node_(3) $udp_(3)
```

```
set null_(3) [new Agent/Null]
$ns_ attach-agent $node_(34) $null_(3)
set cbr_(3) [new Application/Traffic/CBR]
$cbr_(3) set packetSize_ 512
$cbr_(3) set interval_ 0.050000000000000003
$cbr_(3) set random_ 1
$cbr_(3) set maxpkts_ 10000
$cbr_(3) attach-agent $udp_(3)
$ns_ connect $udp_(3) $null_(3)
$ns_ at 55.634230382570173 “$cbr_(3) start”
#
# 17 connecting to 49 at time 29.546173154165118
#
set udp_(4) [new Agent/UDP]
$ns_ attach-agent $node_(17) $udp_(4)
set null_(4) [new Agent/Null]
$ns_ attach-agent $node_(49) $null_(4)
set cbr_(4) [new Application/Traffic/CBR]
$cbr_(4) set packetSize_ 512
$cbr_(4) set interval_ 0.050000000000000003
$cbr_(4) set random_ 1
$cbr_(4) set maxpkts_ 10000
$cbr_(4) attach-agent $udp_(4)
$ns_ connect $udp_(4) $null_(4)
$ns_ at 29.546173154165118 “$cbr_(4) start”
#
#Total sources/connections: 4/5
#
```

APPENDIX C. SAMPLE MOBILITY SCRIPT GENERATED BY BONNMOTION SOFTWARE

```
#====================================================================
# RPGM parameters .. in .param file
#====================================================================

model=RPGM
groupsize_E=15.0
groupsize_S=2.0
pGroupChange=0.1
maxdist=100.0
minspeed=0.0
maxspeed=10.0
maxpause=25.0
ignore=3600.0
randomSeed=1099468043995
x=980.0
y=980.0
duration=200.0
nn=80

#====================================================================
# Sample RPGM mobility file .. in .ns_movements file
#====================================================================

$node_(0) set X_ 629.7970302946021
$node_(0) set Y_ 787.78062043861
$ns_ at 0.0 “$node_(0) setdest 490.2705761867927 783.8517252472469
8.164934979314664”
$ns_ at 17.095268970869256 “$node_(0) setdest 578.0495167815171
792.9678825235802 4.54024357204942”
$ns_ at 36.532782051412596 “$node_(0) setdest 777.3254644721561
168.72112127677858 9.905413954114042”
$ns_ at 102.68673842235285 “$node_(0) setdest 715.0497216127015
155.2864426845972 3.329387541394152”
$ns_ at 121.82190592405095 “$node_(0) setdest 719.9374843614141
408.13292922823473 3.2348412642750963”
$node_(1) set X_ 594.1053051726406
$node_(1) set Y_ 838.6231849437131
$ns_ at 0.0 “$node_(1) setdest 452.9566646908128 867.0531755496078
8.422408755011935”
$ns_ at 17.095268970869256 “$node_(1) setdest 497.2344277331557
837.8667601279535 2.728320016422733”
$ns_ at 36.532782051412596 “$node_(1) setdest 785.0312658309607
264.2715597049949 9.700801889699896”
$ns_ at 102.68673842235285 “$node_(1) setdest 639.0486037464526
205.56028275289106 8.222901484936008”
$ns_ at 121.82190592405095 “$node_(1) setdest 700.6495112944287
433.9610650531058 3.0259374125114955”
$node_(2) set X_ 669.0295628271763
$node_(2) set Y_ 756.5001036035007
$ns_ at 0.0 “$node_(2) setdest 567.5637063041529 779.7065425583435
6.088574834081663”
```

```
$ns_ at 17.095268970869256 “$node_(2) setdest 506.62998032220804
809.0539397606442 3.4794959792495694”
$ns_ at 36.532782051412596 “$node_(2) setdest 728.7798799753147
221.78974154224014 9.491151622975224”
$ns_ at 102.68673842235285 “$node_(2) setdest 643.1932148912763
144.04258378457786 6.042665123356572”
$ns_ at 121.82190592405095 “$node_(2) setdest 703.8697518697142
400.37736611705947 3.3694634153159106”
$node_(3) set X_ 631.4640630984525
$node_(3) set Y_ 793.8749993449281
$ns_ at 0.0 “$node_(3) setdest 499.6010998020829 791.5740157328052
7.714592724813111”
$ns_ at 17.095268970869256 “$node_(3) setdest 501.50517323670084
778.84239703312 0.6622870346768875”
$ns_ at 36.532782051412596 “$node_(3) setdest 708.2744305259733
195.51312802885928 9.355319491085941”
$ns_ at 102.68673842235285 “$node_(3) setdest 701.4723058198073
246.4769892301915 2.686979129449658”
$ns_ at 121.82190592405095 “$node_(3) setdest 750.1094376466106
423.8068994123469 2.3520520741726374”
$node_(4) set X_ 653.8630432569914
$node_(4) set Y_ 751.4624374972367
$ns_ at 0.0 “$node_(4) setdest 516.3899441002852 759.9087182047024
8.05675082926199”
$ns_ at 17.095268970869256 “$node_(4) setdest 538.3597095672555
805.1336750232205 2.586693838085517”
$ns_ at 36.532782051412596 “$node_(4) setdest 543.1354002279188
792.0637285576404 9.521640278370171”
$ns_ at 37.994203113831645 “$node_(4) setdest 642.9419593147421
245.47793094700444 8.588678987937184”
$ns_ at 102.68673842235285 “$node_(4) setdest 682.8454596244079
289.1895550787362 3.093054102672164”
$ns_ at 121.82190592405095 “$node_(4) setdest 706.2187292869976
492.4095473762508 2.6165860467338344”
$node_(5) set X_ 617.9991222994076
$node_(5) set Y_ 747.5057229883114
$ns_ at 0.0 “$node_(5) setdest 467.78595513724713 736.0645307776537
8.812277500420812”
$ns_ at 17.095268970869256 “$node_(5) setdest 465.9445331915517
806.5162297213808 3.625760121888219”
$ns_ at 36.532782051412596 “$node_(5) setdest 735.4601624341481
221.9660347265013 9.73019072491974”
$ns_ at 102.68673842235285 “$node_(5) setdest 657.1661397748132
134.4971819494704 6.134853904364895”
$ns_ at 121.82190592405095 “$node_(5) setdest 712.8837703623976
396.3165137436987 3.4240065456098088”
$node_(6) set X_ 651.9023131794579
$node_(6) set Y_ 797.182171390044
$ns_ at 0.0 “$node_(6) setdest 557.257647191776 843.276909906864
6.158000956895394”
$ns_ at 17.095268970869256 “$node_(6) setdest 491.87001927081525
752.9961419005945 5.734925146917447”
$ns_ at 36.532782051412596 “$node_(6) setdest 706.3090770151543
143.25495533319048 9.770391886431225”
$ns_ at 102.68673842235285 “$node_(6) setdest 736.1562543082782
197.3245658351079 3.2275986587422185”
- cut from original file -
```

APPENDIX D. SAMPLE TRACE FILE FORMAT

```
s 2.556838879 _1_ AGT  --- 0 cbr 512 [0 0 0 0] ------- [1:0 17:0 32 0]
[0] 0 0
r 2.556838879 _1_ RTR  --- 0 cbr 512 [0 0 0 0] ------- [1:0 17:0 32 0]
[0] 0 0
s 2.556838879 _1_ RTR  --- 0 AODV 48 [0 0 0 0] ------- [1:255 -1:255 30
0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
s 2.556953879 _1_ MAC  --- 0 AODV 100 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753895 _6_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753902 _5_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753912 _3_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753926 _0_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753926 _2_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753927 _7_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753929 _10_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753936 _4_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753939 _8_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753968 _11_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557753999 _9_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557754402 _48_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557754707 _47_ MAC  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778895 _6_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778902 _5_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778912 _3_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778926 _0_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778926 _2_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778927 _7_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778929 _10_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778936 _4_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778939 _8_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778968 _11_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557778999 _9_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557779402 _48_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
r 2.557779707 _47_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [1:255
-1:255 30 0] [0x2 1 1 [17 0] [1 4]] (REQUEST)
s 2.558206944 _9_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
s 2.558521944 _9_ MAC  --- 0 AODV 100 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
s 2.558660122 _6_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] ------- [6:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
s 2.558917789 _47_ RTR  --- 0 AODV 48 [0 ffffffff 1 800] -------
[47:255 -1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322013 _11_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322013 _8_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322019 _0_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322046 _10_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322065 _1_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322076 _4_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322079 _6_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322088 _5_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322094 _3_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322110 _7_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322111 _2_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559322503 _48_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347013 _11_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347013 _8_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347019 _0_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347046 _10_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347065 _1_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347076 _4_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347079 _6_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347088 _5_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
r 2.559347094 _3_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] ------- [9:255
-1:255 29 0] [0x2 2 1 [17 0] [1 4]] (REQUEST)
```

APPENDIX E. SAMPLE AWK AND PERL SCRIPTS

PERL SCRIPT for generating simulation scenario and calculation of packet delivery ratio as well as routing overheads.

```
#!/usr/bin/perl

# Generating script for NS2 simulation
# Calculate routing pkts, PDR throughput
# Author: kt Lee

for($k=1;$k<4;$k++)
{
    system “ns wireless_aodv.tcl a$k.tr a$k.nam cbr-50-c5r20-$k
rpgm1.ns “;
    system “ns wireless_dsr.tcl d$k.tr d$k.nam cbr-50-c5r20-$k rpgm1.ns
“;
    system “ns wireless_nrlolsr.tcl ol$k.tr ol$k.nam cbr-50-c5r20-$k
rpgm1.ns “;
    system “ns wireless_dsdv.tcl ds$k.tr ds$k.nam cbr-50-c5r20-$k
rpgm1.ns”;

    system “echo === route/data goodput=======“;
    system “awk -f data.awk a$k.tr >> d_a1”;
    system “awk -f routepkt.awk a$k.tr >> r_a1”;
    system “awk -f data.awk d$k.tr >> d_d1”;
    system “awk -f routepkt.awk d$k.tr >> r_d1”;
    system “awk -f data.awk ds$k.tr >> d_ds1”;
    system “awk -f routepkt.awk ds$k.tr >> r_ds1”;
    system “awk -f data.awk ol$k.tr >> d_ol1”;
    system “awk -f routepkt.awk ol$k.tr >> r_ol1”;

    #system “echo $k”;
    system “rm -f *.nam”;

}
system “mkdir res1”;
system “cp *.awk res1”;
system “mv  *.tr d_* r_* res1”;

system “echo === end of simulation ===“;
```

PERL SCRIPT for generating Simulation Scenario for Energy Consumption Analysis

```
#!/usr/bin/perl

# Generating script for NS2 energy consumption simulation
# Calculate energy consumption
# Author: kt Lee

for($i=1;$i<4;$i++)
    {
    system “ns w_enaodv.tcl ea$i.tr ea$i.nam cbr-50-c5r20-$i
rpgm1.ns”;
```

```
    system “ns w_endsdv.tcl eds$i.tr eds$i.nam cbr-50-c5r20-$i
rpgm1.ns”;
    system “ns w_endsr.tcl ed$i.tr ed$i.nam cbr-50-c5r20-$i rpgm1.ns”;
    system “ns w_ennrlolsr.tcl eol$i.tr eol$i.nam cbr-50-c5r20-$i
rpgm1.ns”;

    system “echo === route/data goodput=======“;
    system “awk -f energy.awk ea$i.tr >> e_a1”;
    system “awk -f energy.awk ed$i.tr >> e_d1”;
    system “awk -f energy.awk eds$i.tr >> e_ds1”;
    system “awk -f energy.awk eol$i.tr >> e_ol1”;

    system “echo $i”;
    system “rm -f e*.nam”;
    }

system “mkdir res1”;
system “mv e*.tr e_* res1”;

system “echo === end of simulation ===“;
```

PERL SCRIPT for calculation of average network delay

```
#!/usr/bin/perl

# Generating script for NS2 simulation
# Calculate delay
# Author: kt Lee

    system “echo =====delay=======“;
    system “echo a1 >>de_a1”;
    system “awk -f delay.awk s=1 d=17 a1.tr >> de_a1”;
    system “awk -f delay.awk s=12 d=25 a1.tr >> de_a1”;
    system “awk -f delay.awk s=4 d=16 a1.tr >> de_a1”;
    system “awk -f delay.awk s=36 d=27 a1.tr >> de_a1”;
    system “awk -f delay.awk s=17 d=20 a1.tr >> de_a1”;

    system “echo d1 >>de_d1”;
    system “awk -f delay.awk s=1 d=17 d1.tr >> de_d1”;
    system “awk -f delay.awk s=12 d=25 d1.tr >> de_d1”;
    system “awk -f delay.awk s=4 d=16 d1.tr >> de_d1”;
    system “awk -f delay.awk s=36 d=27 d1.tr >> de_d1”;
    system “awk -f delay.awk s=17 d=20 d1.tr >> de_d1”;

    system “echo ds1 >>de_ds1”;
    system “awk -f delay.awk s=1 d=17 ds1.tr >> de_ds1”;
    system “awk -f delay.awk s=12 d=25 ds1.tr >> de_ds1”;
    system “awk -f delay.awk s=4 d=16 ds1.tr >> de_ds1”;
    system “awk -f delay.awk s=36 d=27 ds1.tr >> de_ds1”;
    system “awk -f delay.awk s=17 d=20 ds1.tr >> de_ds1”;

    system “echo ol1 >>de_o1”;
    system “awk -f delay.awk s=1 d=17 ol1.tr >> de_o1”;
    system “awk -f delay.awk s=12 d=25 ol1.tr >> de_o1”;
    system “awk -f delay.awk s=4 d=16 ol1.tr >> de_o1”;
    system “awk -f delay.awk s=36 d=27 ol1.tr >> de_o1”;
    system “awk –f delay.awk s=17 d=20 ol1.tr >> de_o1”;

    system “echo =====delay=======“;
    system “echo a2 >>de_a1”;
    system “awk –f delay.awk s=2 d=18 a2.tr >> de_a1”;
    system “awk –f delay.awk s=16 d=18 a2.tr >> de_a1”;
    system “awk –f delay.awk s=34 d=45 a2.tr >> de_a1”;
    system “awk –f delay.awk s=49 d=1 a2.tr >> de_a1”;
    system “awk –f delay.awk s=5 d=21 a2.tr >> de_a1”;
    system “echo d2 >>de_d1”;
    system “awk –f delay.awk s=2 d=18 d2.tr >> de_d1”;
    system “awk –f delay.awk s=16 d=18 d2.tr >> de_d1”;
    system “awk –f delay.awk s=34 d=45 d2.tr >> de_d1”;
    system “awk –f delay.awk s=49 d=1 d2.tr >> de_d1”;
    system “awk –f delay.awk s=5 d=21 d2.tr >> de_d1”;
    system “echo ds2 >>de_ds1”;
    system “awk –f delay.awk s=2 d=18 ds2.tr >> de_ds1”;
    system “awk –f delay.awk s=16 d=18 ds2.tr >> de_ds1”;
    system “awk –f delay.awk s=34 d=45 ds2.tr >> de_ds1”;
    system “awk –f delay.awk s=49 d=1 ds2.tr >> de_ds1”;
    system “awk –f delay.awk s=5 d=21 ds2.tr >> de_ds1”;
    system “echo o2 >>de_o1”;
    system “awk –f delay.awk s=2 d=18 ol2.tr >> de_o1”;
    system “awk –f delay.awk s=16 d=18 ol2.tr >> de_o1”;
    system “awk –f delay.awk s=34 d=45 ol2.tr >> de_o1”;
    system “awk –f delay.awk s=49 d=1 ol2.tr >> de_o1”;
    system “awk –f delay.awk s=5 d=21 ol2.tr >> de_o1”;
```

AWK scripts for packet delivery ratio calculation

```
BEGIN {drop=droppkt=recv=sentpkt=0;}
/^s/ {if(($7==“cbr”)&&($4==“AGT”)){sent++;sentpkt+=$8;}}
/^r/ {if(($7==“cbr”)&&($4==“AGT”)){recv++;recvpkt+=$8;}}
/^D/ {if($7==“cbr”) drop++;droppkt+=$8}
/^f/ {if($7==“cbr”) forw++;forwpkt+=$8}
END {
  #printf(“Dropped pkt: %d and size: %d \n”,drop,droppkt);
  #printf(“Forward pkt: %d and size: %d \n”,forw,forwpkt);
  #printf(“Sent pkt: %d and size: %d \n”,sent,sentpkt);
  #printf(“Received pkt: %d and size: %d \n”,recv,recvpkt);
  goodput=recv/sent;
  goodp=recvpkt/sentpkt;
  #print recv” “sent” “drop” “goodput;
   print “PDR(r/s) [r/s/d]: “goodput*100”% “recv” “sent” “drop”;
}
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[**AWK PERL**] Awk - http://cm.bell-labs.com/cm/cs/awkbook/index.html accessed on 1 July 2004 and PERL - http://www.northbound-train.com/perlwin32.html#Where accessed on 1 July 2004.

[**Broch 1998**] J. Broch, D. Maltz, D. B. Johnson, Yih-Chun Hu, J. Jetcheva. "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols." Proceedings of the Fourth Annual ACM/IEEE on Mobile Computing and Networking, MOBICOM 98, October 1998.

[**Broch 1999b**] Josh Broch, David A. Maltz, and David B. Johnson. "Supporting Hierarchy and Heterogeneous Interfaces in Multi-Hop Wireless Ad Hoc Networks," Proceedings of The International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'99), Workshop on Mobile Computing, Perth, Western Australia, June 1999.

[**Camp 2002**] T. Camp, J Boleng, V. Davies. "A Survey of Mobility Models for Ad Hoc Network Research," Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, 2002, vol. 2, no. 5, pp. 483-502.

[**Chiang 1997**] C.-C. Chiang, H.K.Wu, W. Liu, and M. Gerla. "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel," Proceedings of IEEE SICON'97, April 1997, pp. 197-211.

[**Das 2000**] Samir, Das, Robert, Castañeda, Jiangtao Yan: "Simulation-Based Performance Evaluation of Routing Protocols for Mobile Ad Hoc Networks," Mobile Networks and Applications. May 2000, pp. 179-189, http://www.cc.jyu.fi/~ngerman/HANNIBAL/NNIB/Papers/TP8.pdf acceessed on 18 August 2004.

[**DOCU**] NS2 Documentation Website: http://www.isi.edu/nsnam/ns/ns-documentation.html accessed on 2 June 2004.

[**Gerla 2000**] G. Pei, M. Gerla and T.-W. Chela, "Fisheye State Routing in Mobile Ad Hoc Networks," In Proceedings of the 2000 ICDCS Workshops, Taipei, Taiwan, April 2000, pp. D71-78

[**GLO**] http://pcl.cs.ucla.edu/projects/glomosim/ accessed on 1 May 2004.

[**Guerrero 2001**] Manel Guerrero Zapata. "Secure Ad Hoc On-Demand Distance Vector (SAODV) Routing," IETF Internet Draft, draft-guerrero-manet-saodv-00.txt, August 2001.

[**Haas 2002**] Z. J. Haas, M. R. Pearlman and P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," IETF, MANET Internet Draft, July 2002.

[**Hong 1999**] X. Hong, M. Gerla, G. Pei and C.C. Chiang. "A Group Mobility Model for Ad Hoc Wireless Networks," Proceeding of the 2nd ACM/IEEE Int. Workshop on Modeling and Simulation of Wireless and Mobile Systems (MSWiM'99), 1999, pp. 53-60.

[**Hu 2003**] Yih-Chun Hu, David B. Johnson and Adrian Perrig. "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," Ad Hoc Networks Journal, 1(2003), pp. 175-192.

[**Inter**] Webpage of Inter-4 http://www.inter-4.com accessed on 12 October 2004.

[**Johnson 2000**] David B. Johnson, David A. Maltz, and Josh Broch. "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5, pp. 139-172, Addison-Wesley, 2001.

[**Maltz 1996**] D. B. Johnson, David Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks." Mobile Computing, 1996, pp. 153-181

[**NRL**] NRL is the corporate research laboratory for the Navy and Marine Corps http://www.nrl.navy.mil accessed on 2 June 2004. The OLSR version can be downloaded from:
http://pf.itd.nrl.navy.mil/project/showfiles.php?group_id=16414&release_id=18186
accessed on 2 June 2004.

[**NS2**] Network Simulator -2. http://www.isi.edu/nsnam/ns/ accessed on 1 April 2004.

[**OLSR 2003**] Philippe Jacquet, Paul M¨uhlethaler, Amir Qayyum, Anis Laouiti, Laurent Viennot, Thomas Clausen. "Optimized Link State Routing Protocol." RFC 3626, October 2003.

[**OMN**] http://www.omnetpp.org/ accessed on 1 May 2004.

[**OPNET**] http://www.opnet.com accessed on 1 May 2004.

[**OSPF RFC**] OSPF version 1,2 and 3 RFC website.
http://www.networksorcery.com/enp/protocol/ospf.htm accessed on 1 October 2004.

[**Papa 2002**] Papadimitratos, P. and Haas, Z. "Secure Routing for Mobile Ad Hoc Networks," SCS Communication Networks and Distributed System Modeling and Simulation Conference, 27-31 January 2002, pp. 27-31.

[Pearlman 1999] Pearlman, Marc R., Haas, Zygmunt J. "Determining the Optimal Configuration for the Zone Routing Protocol," IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8. August 1999.

[**Pei 2000**] G. Pei et al. “LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility” International Symposium on Mobile Ad Hoc Networking & Computing, 2000, pp. 11-18.

[**Perkins 1994**] C. E. Perkins and P. Bhagwat “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers,” Computer Communications Review, October 1994, pp. 234-244.

[**Perkins 1999**] C. E. Perkins and E. Royer. “Ad-Hoc On-Demand Distance Vector Routing,” Proceeding of 2nd IEEE Workshop on Mobile Computing & Systems and Applications, February 1999.

[**Perrig 2002**] Yih-Chun Hu, Adrian Perrig, David B. Johnson. “Ariadne: A Secure on-Demand Routing Protocol for Ad Hoc Networks,” Proceedings of the 8th Annual International Conference on Mobile Computing and Networking, 2002, pp. 12-23.

[**QUAL**] http://www.scalable-networks.com/ accessed on 2 June 2004.

[**Ramasubramanian 2003**] V. Ramasubramanian, Z. J. Haas Emin Gün Sirer. “SHARP: A Hybrid Adaptive Routing Protocol for Mobile Ad Hoc Networks,” International Symposium on Mobile Ad Hoc Networking & Computing, 2003, pp. 303-314.

[**RFC 3626**] OLSR RFC: http://www.ietf.org/rfc/rfc3626.txt?number=3626 accessed on 1 May 2004.

[**RIP RFC**] RIP Version 2 RFC Website. http://www.faqs.org/rfcs/rfc2453.html accessed on 1 October 2004.

[**Royer 1999**] Elizabeth M. Royer & C. K. Toh. “A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks.” IEEE Personal Communications, April 1999.

[**Rugged**] Webpage of Rugged Notebooks http://www.ruggednotebooks.com/ accessed on 12 October 2004.

[**Tech 1998**] “Selection Procedures for the Choice of Radio Transmission Technologies of the UMTS.” Technical Report tr 101 112 v3.2.0, ETSI SMG-5, France, April 1998.

[**Througput**] http://asia.cnet.com/reviews/hardware/networking/0,39001739,39150884-2,00.htm accessed on 22 November 2004.

[**Toh 1996**] C. K. Toh. “A Novel Distributed Routing Protocol to Support Ad-Hoc Mobile Computing,” Proceeding 1996 IEEE 15th Int’l Phoenix Conference Computing and Communication. March 1996, pp. 480-86.

[**TRACE**] Old and New Trace Format for NS2 Logged Trace Files. http://k-lug.org/~griswold/NS2/ns2-trace-formats.html accessed on 2 July 2004.

[**Tsuchiya 1988**] E. E. Tsuchiya, "The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks," In Computer Communication Review*,* vol. 18, no.4, August 1988, pp. 35-42.

[**Waal 2003**] Christian de Waal and Michael Gerharz. "BonnMotion: A Mobility Scenario Generation and Analysis Tool." Communication Systems Group, Institute of Computer Science IV, University of Bonn, Germany. Website: http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/ accessed on 10 June 2004.

[**Yi 2001**] Seung Yi, Prasad Naldurg, Robin Kravets. "A Security-Aware Routing Protocol for Wireless Ad Hoc Networks," Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing, 2001, pp. 299-302.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor Geoffrey Xie
   Naval Postgraduate School
   Monterey, California

4. Professor Su Wen
   Naval Postgraduate School
   Monterey, California

5. Sim Siew See
   Defence Science & Technology Agency
   Singapore

6. Yeo Tat Soon
   Temasek Defense System Institute
   National University of Singapore
   Singapore

7. Lee Kok Thong
   Defence Science & Technology Agency
   Singapore